EXELOGO

MANUEL D'INITIATION
MANUEL DE REFERENCE



LE LOGICIEL EXELOGO A ETE DEVELOPPE
PAR ACT INFORMATIQUE ET EXELVISION

L'ensemble EXELOGO a été réalisé par les équipes de:

ACT Informatique et EXELVISION.

La partie logicielle a été effectuée pour ACT sous la conduite de :
Armen VALIAN.
La rédaction du présent ouvrage a été réalisée par:
Doris AVRAM et Gérard DAIIAN

Pour Exelvision les personnes qui ont collaboré sont:

Patrick DUPLOUY : Système, éditeur.
Alain MARAFETTI : Le graphisme, les lutins.
Jean Michel GAUSSEN : Les entrées / sorties , les routines mathématiques.
Patrice CHAILLAN : Rédaction, adaptation documentation et mise en page.

L'équipe d'EXELVISION tient à remercier l'équipe d'ACT pour leur accueil et leurs nuits fébriles.

Avertissement.

Le logiciel EXELOGO ainsi que ces ouvrages sont protégés par un copyright de ACT informatique, PARIS . La garantie offerte par ACT informatique se limite à la garantie exprimée dans les termes des articles 1641 et suivants du code civil français.

ACT informatique a fait tous les efforts pour assurer la validité la meilleure du logiciel informatique ainsi que cet ouvrage à la date de parution. ACT informatique ne saurait cependant ni être tenu pour responsable des conséquences éventuelles directes ou indirectes que pourraient entraîner de telles améliorations ou un usage non conforme à la destination du logiciel EXELOGO.

©ACT informatique
12, rue de la montagne Ste Geneviève
Paris 75005, France

Paris 1985

Toute reproduction intégrale ou partielle sans le consentement de l'auteur ou de ses ayants-droits, est illicite (loi du 11 Mars 1957, alinéa 1 de l'article 40) .
Cette reproduction intégrale, par quelque procédé que se soit, constituerait une contrefaçon sanctionnée par les articles 425 et suivants du code Pénal.
Toute reproduction au mépris de ces textes, de tout ou partie de cette documentation, entrainerait de facto, la reproduction de tout ou partie du programme associé.

Introduction

Félicitations! Vous venez d'acquérir EXELOGO, un langage très complet et parfaitement adapté à l'ordinateur EXL 100.

Accessible au jeune débutant, ce langage est tout aussi riche pour stimuler et satisfaire un programmeur averti. Logo est le fruit de l'Intelligence Artificielle et de la Psychologie Cognitive, créé dans les laboratoires du MIT par l'équipe de Seymour Papert.

EXELOGO a été développé par ACT Informatique, un des premiers spécialistes français en Logo et langages de l'Intelligence Artificielle et la Société EXELVISION: il est compatible avec les normes européennes de Logo en langue française, celles de l'Education Nationale française et de cantons francophones hélvétiques. Il a été réalisé pour tous les ordinateurs présents dans les systèmes éducatifs.

Ce manuel vous initiera à la programmation en Logo d'une façon originale.

Sur les pages de gauche vous trouverez des exemples et des exercices que vous pourrez taper vous-mêmes directement au clavier. Sur les pages de droite vous trouverez l'explication des principes du langage.

Si vous voulez apprendre en utilisant des exemples, commencez par les pages de gauche et si vous préférez comprendre le fonctionnement du langage avant de vous lancer à des réalisations, plongez-vous dans les pages de droite. A vous de choisir !

Evidemment, les pages de droite et de gauche même si elles ne sont par directement liées,recouvrent deux aspects d'un même thème, le thème du chapitre.

Dans les 6 premiers chapitres de ce manuel, sont introduites les idées fondamentales de la programmation en Logo par le graphique. A partir du chapitre 9 vous découvrirez l'utilisation de l'arithmétique et du texte. Les chapitres 4 et 5 portent sur la manipulation de l'éditeur et de la mémoire.

Le **manuel d'initiation** vous donne un premier aperçu du langage. Le **manuel de référence** vous permettra d'approfondir ses possibilités. A vous de découvrir sa richesse.

Alors, chargez EXELOGO dans votre ordinateur, tournez cette page et allez-y!...

Mise en route du système

Pour charger Exelogo, introduisez la cartouche de programme dans la trappe prévue à cet effet. Ensuite, allumez votre ordinateur et votre poste de télévision. Dès que le message BIENVENUE A LOGO apparaît, vous pouvez commencer...

Chapitre 1

La tortue en direct

Nous commencerons l'apprentissage du langage Logo en dessinant avec la célèbre tortue. Ne croyez pas par là à un attachement excessif de notre part pour cet animal! Tout simplement, cette tortue est un bon moyen pour comprendre les idées fondamentales de la programmation en Logo.

Voici la tortue!

Quand on commande:
MT
Un petit papillon apparaît au milieu de l'écran. C'est la TORTUE! (Nous employons le papillon à la place de la tortue car le papillon est la mascotte d'EXELVISION). Elle sera votre guide dans vos premiers pas en Logo.

Cette tortue peut bouger quand vous le lui demandez. Elle se déplace, ou pivote. Vous serez surpris par le nombre de réalisations possibles juste avec ces deux mouvements.

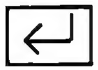
Pour la faire bouger, faites attention à son **orientation**, vers où elle regarde; et à sa **position**, où elle se trouve sur l'écran. Ces deux éléments donnent l'**état** de la tortue. Observez toujours son état, cela vous évitera des ennuis....

La tortue a aussi un **crayon**. S'il est baissé, elle laisse la trace de son passage, s'il est levé elle ne laissera pas de trait, s'il se transforme en gomme elle effacera le trait sur lequel elle passe, s'il est de couleur....

Commandes de base pour la tortue.

Au début, la tortue comprend seulement certains ordres. Nous présentons quelques uns dans ce chapitre. Tapez vos ordres sur le clavier et dès que vous appuierez sur la touche *(ENTREE*)*, elle les exécutera.

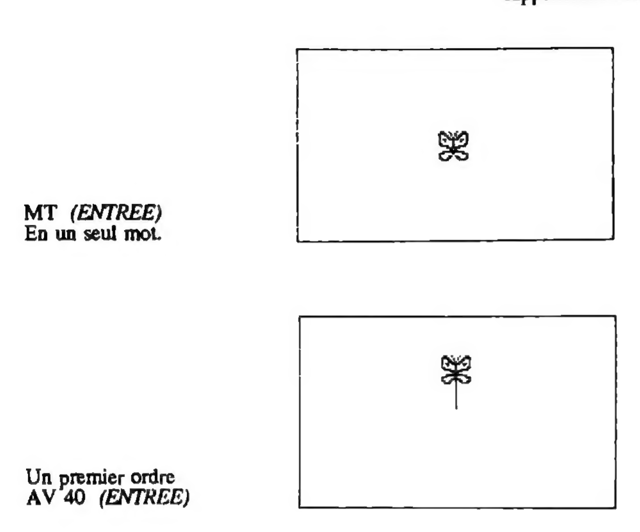
Méfiez-vous, la tortue est bête et disciplinée. Si elle n'a pas fait ce **que vous pensiez,** elle a fait **ce que vous lui avez demandé.** Il faut vous exprimer avec son vocabulaire.

* La touche ENTREE est représentée sur votre clavier par la touche 

Le point de départ!

Appelons la tortue.

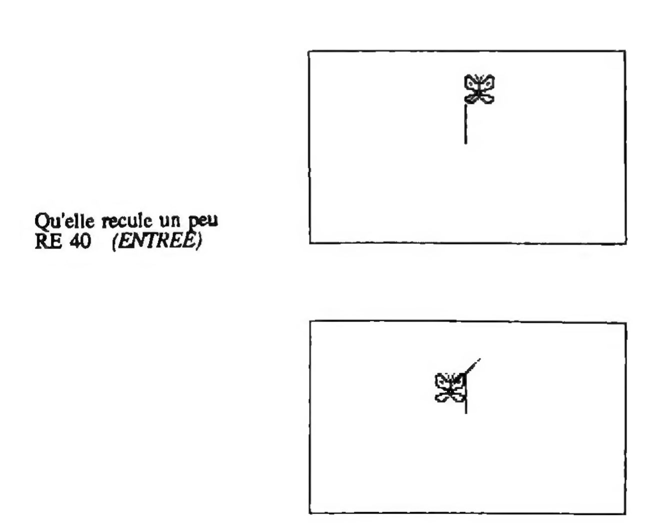
Aprés avoir tapé vos instructions, appuyez sur *(ENTREE)*. Nous vous rappellerons cette action dans ce chapitre.



MT *(ENTREE)*
En un seul mot.

Un premier ordre
AV 40 *(ENTREE)*

Faisons-la tourner
TD 45 *(ENTREE)*
AV 20 *(ENTREE)*
L'ordre s'écrit en un mot, mais un espace doit être mis entre l'ordre et sa donnée (le nombre).



Qu'elle recule un peu
RE 40 *(ENTREE)*

Les ordres connus au départ s'appellent des **primitives**. En voici quelques unes:

AV ou AVANCE
RE ou RECULE
TD qui signifie TOURNEDROITE
TG qui signifie TOURNEGAUCHE

Chacune de ces primitives doit être accompagnée d'une information appelée **donnée.** Il faut préciser de combien de pas la tortue doit avancer (ou reculer) et de combien de degrés elle doit tourner (à droite ou à gauche). Pour la faire tourner, rappelez-vous qu'un tour complet est de 360 degrés.

Certaines primitives ont **toujours besoin d'une donnée.** Vous venez de voir 4 d'entre elles. Quand vous les utiliserez, n'oubliez pas de mettre la donnée en la séparant bien par un espace (barre d'espacement). Comparez :

AVANCE45TD90RECULE56 avec AV 45 TD 90 RECULE 56

Si la première suite d'ordres est pour vous difficile à lire, pour l'ordinateur elle est incompréhensible.

D'autres primitives n'ont **pas besoin de données.**
Par exemple:

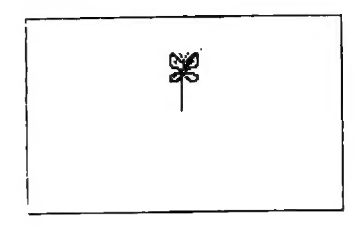
MT qui signifie MONTRETORTUE
CT qui signifie CACHETORTUE
BC qui signifie BAISSECRAYON
LC qui signifie LEVECRAYON
GC qui signifie GOMMECRAYON
VE qui signifie VIDECRAN pour vider l'écran et remettre la tortue au départ.

L'exécution des commandes.

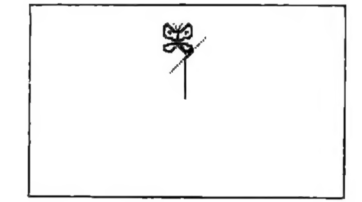
Voyez-vous le point d'interrogation (?) en début de ligne? On l'appelle "symbole d'invite" car il vous invite à taper des instructions. Dès que vous les aurez tapées, pensez à appuyer sur la touche *(ENTREE)* pour que l'ordinateur les exécute.

Plusieurs commandes sur une ligne

AV 50 RE 20 *(ENTREE)*

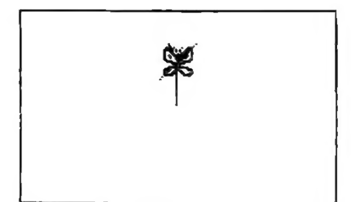


TG 75 AV 20 *(ENTREE)*



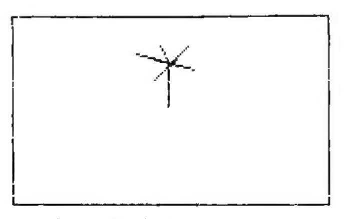
Continuons

RE 20 *(ENTREE)*



Encore un peu...

TG 45 *(ENTREE)*



AV 35 RE 60 CT *(ENTREE)*
Et voilà!
Faites mieux. Recommencez en faisant toutes les branches de l'étoile égales.

Ce mode de fonctionnement s'appelle le **mode direct**. Quand le symbole d'invite en début de ligne est (?) la touche *(ENTREE)* conclut la **ligne Logo*** et "l'envoie" à l'ordinateur. C'est comme si vous lui disiez à ce moment "Maintenant, fais ce que je demande!". L'ordinateur, lui, s'empressera d'exécuter vos ordres. S'il ne peut pas, il vous enverra un message Logo.

Les "messages Logo".

Parfois, l'ordinateur ne peut exécuter votre demande. Il affiche alors un message au lieu de faire ce que vous vouliez! Lisez-le attentivement et regardez les instructions que vous venez de lui donner. Ces messages sont votre aide. Voici quelques cas qui arrivent souvent au début:

AVANCE
PAS ASSEZ DE DONNEES POUR AV

Le nombre de pas dont la tortue doit avancer n'a pas été indiqué.

98
QUE FAIRE DE 98

Un nombre est la donnée d'un ordre : avancer, tourner, écrire... vous n'avez pas donné d'ordre à l'ordinateur.

RE98
COMMENT FAIRE POUR RE98

Entre un ordre et sa donnée il faut laisser un espace (barre d'espacement). L'ordinateur considère RECULE98 comme une autre instruction, inconnue.

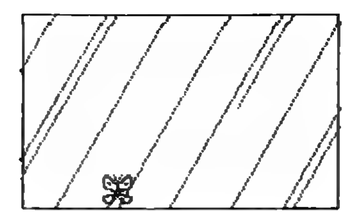
AV TD 45
PAS ASSEZ DE DONNEES POUR AV

Vous vouliez que la tortue avance en tournant à droite de 45 degrés? Eh bien, non! Chaque instruction doit être complète, c'est-à-dire, l'ordre avec sa donnée. AVANCE attend un nombre pour que la tortue puisse avancer.

Dans tous les cas, il faut réécrire correctement les instructions sur la prochaine ligne.

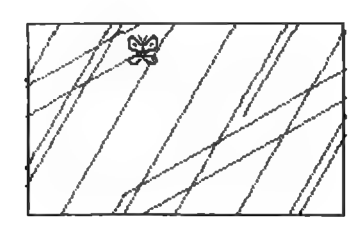
*une Ligne Logo peut contenir beaucoup d'instructions et s'étendre sur plusieurs lignes écran. Elle se termine par l'appui de la touche *(ENTREE)*.

Quelques surprises!!



Pour commencer
VE *(ENTREE)*
MT *(ENTREE)*

TD 30 AV 3600 *(ENTREE)*
Hein?? La tortue sort d'un bord de l'écran et réapparait par le coté opposé. L'écran est enroulé sur lui même, comme si on enroulait un ruban autour d'un pneu. C'est ce que les mathématiciens appellent un TORE.



TD 30 *(ENTREE)*
AV 1500 *(ENTREE)*

Une idée : Aprés avoir fait des zébrures comme celles-ci, ne videz pas l'écran mais continuez à faire des zébrures en gommant.

Les erreurs et comment s'en sortir.

Il est fréquent de faire des erreurs. Mais comme le dit un proverbe bien connu: "Il n'y a que ceux qui restent assis qui ne tombent pas!".

Attention aux fautes de frappe : ne confondez pas la lettre I avec le chiffre 1, ou la lettre O avec zéro. Même s'ils vous semblent similaires, ces caractères envoient des signaux différents à la machine.

Lorsque vous tapez sur le clavier, le carré lumineux qui se déplace sur l'écran est le **curseur**. Il indique la position du prochain caractère tapé.

Certaines touches permettent de déplacer le curseur pour corriger une erreur sur la ligne même que vous tapez.

- Vous n'avez pas encore appuyé sur la touche ***(ENTREE)***

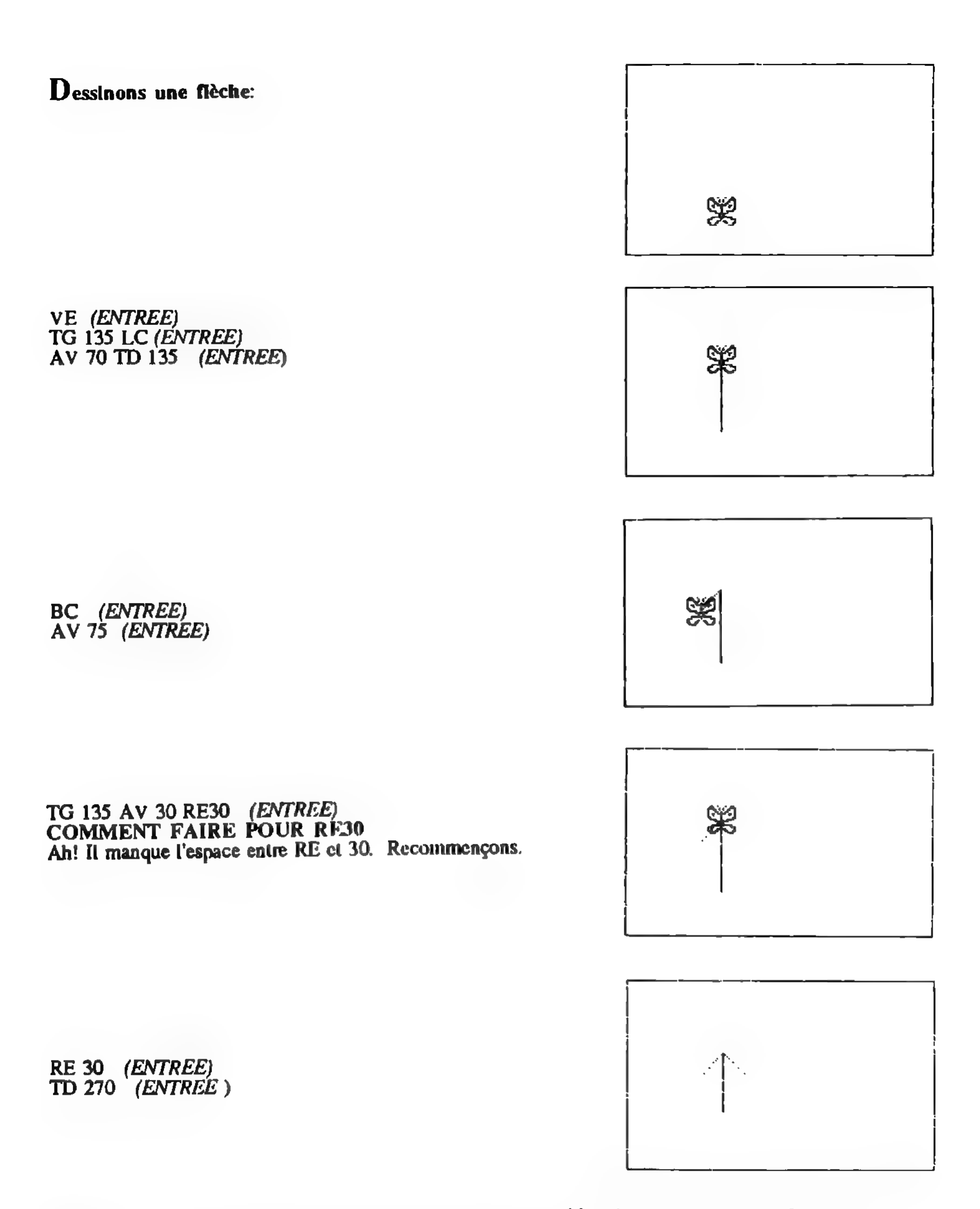
(< X>) Déplace le curseur vers l'arrière en effaçant le caractère. Cette touche est la touche DELETE

- Vous avez déjà appuyé sur ***(ENTREE)***

Trop tard! Le message Logo apparaît. Ignorez ce qu'il y a sur l'écran et retapez correctement les instructions sur la prochaine ligne.

Pour ceux qui ne veulent pas tout retaper :
<CTL R> reproduit, à la position du curseur la ligne précédente.

Dessinons une flèche:



VE *(ENTREE)*
TG 135 LC *(ENTREE)*
AV 70 TD 135 *(ENTREE)*

BC *(ENTREE)*
AV 75 *(ENTREE)*

TG 135 AV 30 RE30 *(ENTREE)*
COMMENT FAIRE POUR RE30
Ah! Il manque l'espace entre RE et 30. Recommençons.

RE 30 *(ENTREE)*
TD 270 *(ENTREE)*

AV 30 CT *(ENTREE)* A propos...pouvez vous trouver combien de pas fait l'écran en hauteur et en largeur?

Chapitre 2

De nouveaux trucs

Moins d'efforts pour faire la même chose! Voilà ce que nous verrons dans ce chapitre. Et si nous n'avons pas la couleur dans ce manuel, vous l'avez peut être sur votre écran. Alors utilisez-la pour rendre vos graphiques plus attrayants.

Plus d'actions en moins d'instructions.

S'il faut répéter plusieurs fois les mêmes actions, il n'est pas besoin de les retaper à chaque fois! Dites à l'ordinateur de le faire par lui-même. La primitive REPETE est là pour ça!

Vous êtes d'accord qu'il est plus long de taper :

AVANCE 50 TD 90
AVANCE 50 TD 90
AVANCE 50 TD 90
AVANCE 50 TD 90

que de taper:

REPETE 4 [AVANCE 50 TD 90]
Avec cette dernière instruction, l'ordinateur exécute par lui-même 4 fois les ordres écrits entre crochets.

Vous rencontrerez souvent les crochets en Logo. Ils délimitent une **liste** : une liste de mots, de nombres, et même une liste de listes.

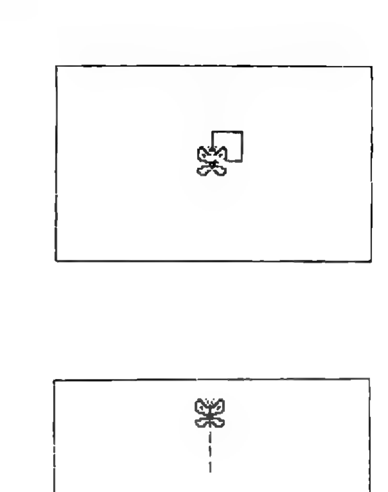
La primitive REPETE reçoit deux données. La première est un nombre, la deuxième est une liste d'instructions. Par exemple:

REPETE 5 [AVANCE 40] équivaut à AVANCE 200
et
REPETE 3 [AVANCE 100 TD 120] tracera une figure connue.

Logo incite toujours à réflechir pour décomposer un problème en unités de base. Quand vous voulez tracer une figure complexe, cherchez s'il y a des éléments qui se répètent.

Des REPETE partout!!

Pensez à Vider l'Ecran entre deux images.



REPETE 4 [AV 50 TD 90]

REPETE 5 [AV 10 LC AV 5 BC]

REPETE 18 [AV 50 RE 50 TD 20]
RECULE 70 CT

REPETE 5 [AV 15 TD 90 AV 15 TG 90]

De la couleur!

Profitez de la couleur que peut offrir votre ordinateur.

Chaque couleur a un code numérique. Vous l'utiliserez comme donnée des primitives qui permettent de changer la couleur du graphique.

Voici quelques primitives pour colorer votre graphique :

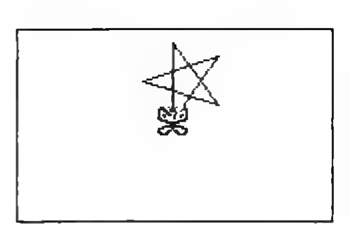
FCC, fixe la couleur du crayon de la tortue à celle que vous précisez. FCC 2 par exemple donne à la tortue un crayon vert . Si le crayon est baissé, la tortue laissera des traces de cette couleur jusqu'à ce que vous modifiez la couleur de son crayon par un autre FCC.

FCFG fixe la couleur du fond graphique à celle que vous avez choisie et indiquée. FCFG 6 par exemple, colorie tout l'écran en CYAN.

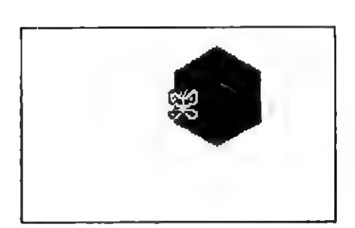
Attention! si la couleur du fond est la même que celle du crayon c'est comme si vous écriviez avec une craie blanche sur du papier blanc: vous ne verrez rien.

Il existe une autre primitive intéressante, REMPLIS.

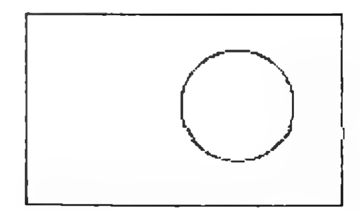
FCFG 4 FCC 6
REPETE 5 [AV 80 TD 144]



REPETE 6 [AV 50 TD 60]
C'est un hexagone.
LC TD 30 AV 20 BC REMPLIS



Remplacez les ?? par des nombres pour obtenir un pentagone.
REPETE 5 [AV 60 TD ??]
Un pentagone a 5 cotés.



CT le tracé se fera plus rapidement.
REPETE 360 [AV 1 TD 1]

Chapitre 3

Comment écrire des procédures

Les dessins faits jusqu'à présent ne durent pas longtemps. L'ordinateur "oublie" les instructions qu'on lui donne dès qu'il les exécute : l'écran vidé, le travail est à refaire.

Nous verrons maintenant comment faire pour que l'ordinateur apprenne et se souvienne de ce qu'il fait. Pour cela, il faut définir des procédures. Quand vous aurez défini des procédures, n'éteignez pas tout de suite l'ordinateur. Regardez d'abord le chapitre 4 pour savoir comment les enregistrer et les ramener ensuite en mémoire.

Les procédures.

En Logo, on peut créer de nouveaux ordres. On les définit avec des mots que l'ordinateur connaît déjà. Ces ordres s'ajoutent au vocabulaire de base compris par l'ordinateur et s'appellent des procédures.

Voici l'exemple d'une définition de procédure, une procédure que nous appelerons TRAJET.

POUR TRAJET........**le titre:** POUR et le nom de la procédure (en un mot)
AVANCE 80 TD 90
AVANCE 80 TD 90 **le corps :** lignes d'instructions
AVANCE 80 TD 90
AVANCE 80 TD 90
FIN.......................**dernière ligne:** uniquement le mot FIN

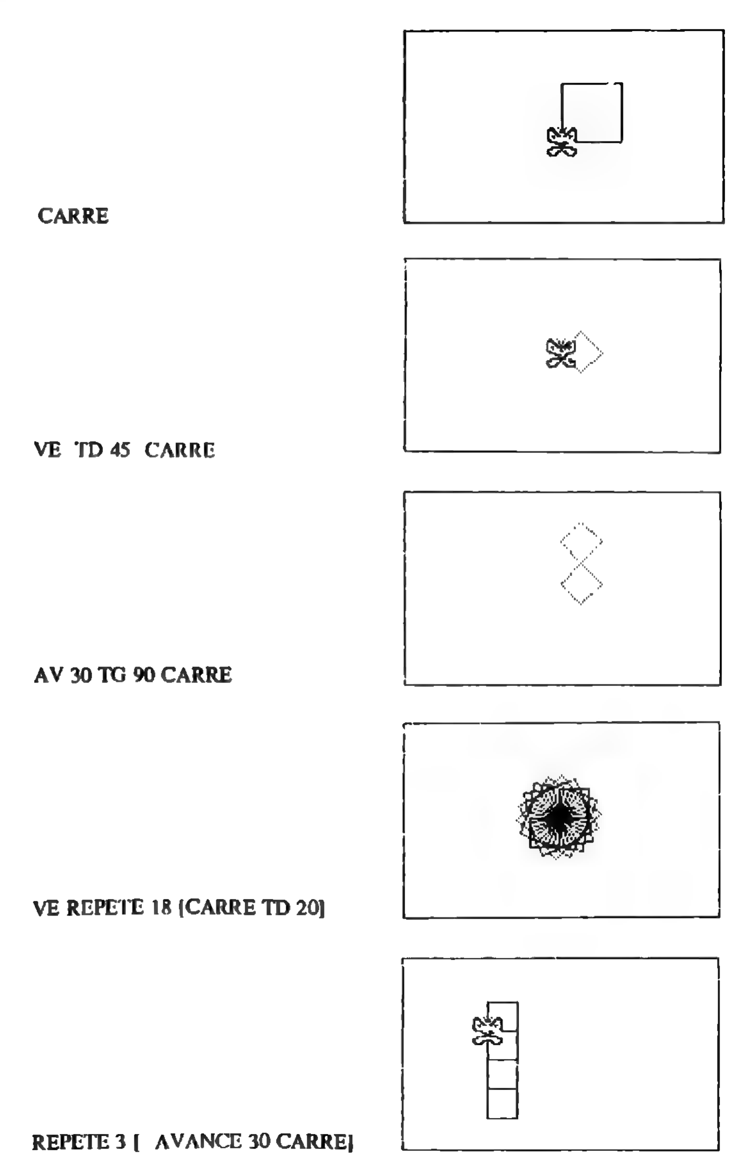
En écrivant cette petite procédure, on a "appris" à l'ordinateur comment faire TRAJET. La preuve : quand l'ordre TRAJET sera donné (en tapant TRAJET suivi d'*(ENTREE)*), il exécutera la suite d'instructions enseignées. TRAJET est maintenant un mot que l'ordinateur connaît. Comme BC ou d'autres encore. Vous pourrez l'utiliser quand vous voudrez.

Mode direct, mode de définition des procédures.

Quand vous êtes en **mode direct** le point d'interrogation est en début de ligne et en appuyant sur *(ENTREE)* l'ordinateur exécute votre ordre.

Tout ce que vous pouvez faire avec un carré....

```
POUR CARRE
REPETE 4 [AV 30 TD 90]
FIN
```



CARRE

VE TD 45 CARRE

AV 30 TG 90 CARRE

VE REPETE 18 [CARRE TD 20]

REPETE 3 [AVANCE 30 CARRE]

Pouvez-vous faire des carrés alignés à l'horizontale?

Lorsque vous définissez une procédure, vous entrez dans le mode de **définition des procédures.**
Après avoir tapé la première ligne (par exemple POUR TRAJET) et appuyé sur *(ENTREE)*, regardez bien le signe qui s'affiche en début de ligne. Au lieu du symbole d'invite habituel (?) vous verrez le signe >.

Tant que le signe > apparaît en début de ligne, l'ordinateur n'exécute pas vos ordres. Il les mémorise car il est en "état d'écoute", et non d'exécution. Ces deux modes sont bien distincts en Logo : tachez de ne pas les confondre.

Dès que vous tapez le mot FIN, et appuyez sur *(ENTREE)*, apparaît le message:
VOUS VENEZ DE DEFINIR TRAJET
et le symbole d'invite du mode direct (?) est à nouveau là.

Vous pouvez alors, demander à l'ordinateur d'exécuter des ordres en les tapant comme d'habitude.

Corriger une procédure.

Pour corriger un erreur dans une procédure deux cas peuvent se présenter :

a. Vous remarquez l'erreur pendant la définition de la procédure.

Si vous n'avez pas encore appuyé sur *(ENTREE)*, déplacez le curseur comme d'habitude pour corriger la faute.

Si vous avez déjà appuyé sur *(ENTREE)*, deux solutions:
- Continuez la définition en ignorant la faute. Après avoir tapé FIN revenez au mode direct (en appuyant sur *(ENTREE)*). Regardez dans le chapitre 5 comment corriger une procédure dans l'éditeur.

- Tapez FIN sur la prochaine ligne, pour retrouver le mode direct. Ensuite, tapez :

EF "TRAJET
pour que l'ordinateur efface cette procédure de sa mémoire. N'oubliez pas de mettre un guillemet devant le nom de la procédure (sans laisser d'espace), vous en trouverez une explication plus tard, dans le chapitre 9 .
Ensuite retapez la procédure en essayant de ne plus faire de fautes.

Les arcs sont toujours utiles....

```
POUR ARC
REPETE 45  [AV 1 TD 2 ]
FIN
```

ARC

....Pour faire un pétale

CT ARC TD 90 ARC
Trouvez la suite pour en faire une fleur..

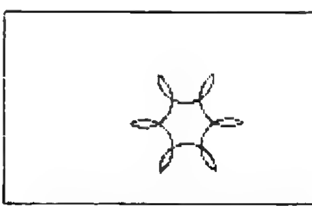
....Pour faire une guirlande

TD 45 REPETE 3 [ARC TG 90]

....Pour faire une rosace

à la bonne place..

LC RECULE 50 BC
REPETE 6 [REPETE 2 [ARC] TD 120]



....Pour faire autre chose avec des arcs....à vous d'imaginer.

b.Vous remarquez l'erreur après l'exécution de la procédure.

Vous avez défini une procédure et à la demande d'exécution vous n'avez pas obtenu ce que vous vouliez! L'ordinateur a pourtant suivi vos instructions à la lettre. Il faudra les revoir! Pour corriger une procédure, il faut passer par l'éditeur. Regardez le chapitre 5.

Autre solution: effacer la procédure de la mémoire,

EF "TRAJET

pour la redéfinir (ce qui revient au même que pour le paragraphe précédent).

Certaines personnes préfèrent ne pas effacer la procédure de la mémoire, mais la redéfinir en l'appelant autrement, par exemple, TRAJETBIS. Ainsi ils gardent en mémoire les deux versions : la première (TRAJET) et la version améliorée (TRAJETBIS).

Logo et les procédures.

Les procédures sont un outil fondamental de Logo. Vous définissez par vos propres mots de nouvelles actions et ainsi, vous enrichissez de plus en plus le langage dont vous disposez.

Souvent, pour les débutants en Logo, les procédures paraissent surtout pratiques car elles restent en mémoire en tout cas tant que l'ordinateur est allumé. Le travail fourni n'est pas perdu car une procédure peut être enregistrée.

Tout un ensemble d'instructions sont ainsi résumées par un seul mot. Pour reproduire leur effet, il n'est pas besoin de les retaper, il suffit simplement de donner l'ordre par lequel elles sont appellées. Une procédure est une "unité" que l'on peut, comme les primitives, utiliser dans la construction d'autres procédures. Comme nous verrons dans le chapitre 6, l'imbrication de procédures est une idée très puissante du langage.

En ce moment vous vous demanderez peut-être : quand commencerons-nous à programmer? La réponse est simple : c'est déjà fait! Un programme en Logo, est un ensemble de procédures et/ou primitives. On dit d'ailleurs souvent "procédure primitive" pour parler des primitives, et "procédure utilisateur" quand il s'agit des procédures.

Il n'y a plus besoin de se rappeler des codes de couleur.

Les procédures peuvent être tellement pratiques quand on n'arrive pas à se souvenir!..

```
POUR CRAYONROUGE
FCC 1
FIN

POUR CRAYONVERT
FCC 2
FIN
```

Si on demande CRAYONROUGE CARRE
le dessin se fait automatiquement en rouge!

Et si on demande CARRE CRAYONVERT
pouvez vous prédire la couleur du dessin?

Vers où regarde la tortue?
Gardez cette procédure toujours en mémoire, et tapez TORTUE dès que vous vous sentez perdus!

```
POUR TORTUE
LC
REPETE 3 [AV 10 ATTENDS 10 RE 10]......ATTENDS fait attendre n parties de seconde
BC                                       entre deux actions.
FIN
```

Avant d'éteindre l'ordinateur, regardez vite le chapitre 4 pour voir comment garder ce que vous venez de faire sur disquette.

Chapitre 4

Un peu de mémoire

Qu'avez vous à votre disposition dans la mémoire de l'ordinateur? Comment garder ce que vous avez déjà fait? Comment le retrouver lors d'une séance ultérieure? Autant de questions auxquelles nous répondons dans ce chapitre. Vous trouverez plus d'informations dans les chapitres du manuel de référence sur la gestion de l'espace de travail et des fichiers.

L'espace de travail.

Nous appelons **espace de travail** tout ce qui se trouve en mémoire centrale au moment où vous travaillez avec l'ordinateur. C'est-à-dire, tout ce qu'il connaissait au départ, et tout ce que vous lui avez appris .

Vous avez dû remarquer qu'il n'est pas possible de définir une procédure en utilisant le nom de celle qui existe déjà car l'ordinateur répond: .....EXISTE DEJA. L'espace de travail ne peut pas contenir deux procédures de même nom et il faut soit donner un autre nom à la nouvelle procédure, soit effacer l'ancienne.

Certaines primitives, comme IMTS ou IM permettent d'examiner ce qu'il y a dans l'espace de travail. D'autres, comme EF (qui signifie EFFACE) ou .EFTOUT permettent d'effacer des éléments de la mémoire centrale.

Cette mémoire fonctionne tant que l'appareil est allumé. Dès qu'il est éteint, tout se perd. Pour pouvoir retrouver son contenu, il faut l'enregistrer. Lors d'une prochaine séance, vous pourrez retrouver ce qui a déjà été fait et continuer pour le perfectionner, l'utiliser dans d'autres constructions ou simplement le consulter.

Si vous venez de terminer le chapitre 3, jetons un coup d'oeil dans l'espace de travail.

IMTS
POUR TORTUE
POUR CRAYONVERT
POUR CRAYONROUGE
POUR ARC
POUR CARRE

Regardons le texte de CARRE

IM "CARRE — N'oubliez pas le guillemet.
POUR CARRE
REPETE 4 [AV 50 TD 90]
FIN

Si on écrivait
IM CARRE — sans le guillemet
PAS ASSEZ DE DONNEES POUR IM

le carré est dessiné

Disons-le clairement : Si on ne met rien devant un mot l'ordinateur pense que c'est un ordre à exécuter.

Effaçons une procédure :

EF "CRAYONROUGE

IM "CRAYONROUGE
CRAYONROUGE N'EXISTE PAS

Evidemment, puisque on vient de l'effacer. Si votre ordinateur ne donne aucun message, vous êtes dans le même cas. Il n'y a pas de procédure à afficher.

Définissons une nouvelle procédure

POUR FONDBLEU
FCFG 4
FIN

Enregistrer.
Tout ce que vous avez créé peut être enregistré.

Pour déterminer sur quel périphérique sera stockée l'information, vous devrez taper:

?EC LECTEUR

LECTEUR retournera une valeur comprise entre 0 et 3. Les codes des périphériques sont les suivants:

-0 ---> Lecteur de cassettes
-1 ---> Exelmémoire
-2 ---> Unité de disquettes A
-3 ---> Unité de disquettes B

Nous supposerons que nous travaillons sur un lecteur de disquettes. Pour plus d'information sur les possibilités de sauvegarde et de chargement, voir le manuel de référence. Pour travailler sur le lecteur de disquettes A, nous avons tapé :

FLECTEUR 2 *(ENTREE)*

La disquette contient des fichiers. Chaque fichier possède un nom et pour mettre votre travail en réserve, il faut donner un nom au fichier. Vous taperez par exemple la ligne:

SAUVE "MARDI13.LOG [CARRE PENTAGONE TOURNOI FLEUR TRIANGLE CERCLE]

Toutes les procédures dont le nom est écrit entre crochets seront enregistrées dans le fichier nommé MARDI13. Le nom du fichier peut être n'importe quel mot*, mais il doit être un seul mot sans espace entre le jour et son quantième, et il ne doit pas avoir plus de 13 caractères. **N'oubliez pas de mettre le guillemet au début du nom sans laisser d'espace.**

En appuyant sur *(ENTREE)* la lumière du lecteur s'allumera. Enregistement en cours! Attendez qu'elle s'éteigne.
Dès que le point d'interrogation et le curseur réapparaissent la sauvegarde est terminée. Vous pouvez enlever la disquette du lecteur et la ranger dans sa pochette. Si vous voulez arrêter de travailler, éteignez l'ordinateur, et...bonne nuit.

*Le nom du fichier est indépendant des noms des procédures. Ainsi, pour se rappeler du contenu d'un fichier, on peut lui donner comme nom le nom d'une des procédures qu'il contient.

Et dans l'ensemble de l'espace de travail....

IMTS
POUR FONDBLEU
POUR TORTUE
POUR CRAYONVERT
POUR ARC
POUR CARRE

Si une unité de disquettes est connectée, faites FLECTEUR 2

Introduisons une disquette dans le lecteur.

Attention! Est-elle initialisée???

Sauvons les procédures
FLECTEUR 2

SAUVE "CHAP4 [FONDBLEU TORTUE CRAYONVERT ARC CARRE]

Guettez le symbole d'invite (?) et le curseur. La sauvegarde est terminée.

Pour sauver

SAUVE "NOMFICHIER [TRAC TARC TCRA TCAR]

Regardons si le nouveau fichier apparait sur la disquette

.SYSTEME (ENTREE)
DIR (ENTREE)

Effaçons tout de la mémoire:

.EFTOUT
IMTS

Rien ne s'affiche car il n'y a plus de procédures en mémoire. Tout a été effacé : il ne reste plus que les primitives.

Ramener en mémoire.

Pour retrouver un travail déjà effectué, après avoir chargé le langage Logo, introduisez votre disquette dans le lecteur, et tapez :

.SYSTEME et ensuite DIR

Peu après avoir appuyé sur *(ENTREE)* s'affichent des informations présentées en colonnes. Par exemple :
MARDI13.LOG
DERIVE
FORMES

La première colonne indique le nom du fichier. La deuxième son extension, c'est-à-dire le type de fichier dont il s'agit. (Notez que vous pouvez enregistrer sur la même disquette d'autres fichiers que du Logo). La troisième colonne indique les protections qui sont appliquées au fichier. La quatrième colonne indique le nombre d'AUs occupées par un fichier (1 AU = 512 octets) . La cinquième colonne indique le nombre maximal d'octets alloués pour un enregistrement. La sixième colonne indique le nombre d'enregistrements du fichier.

Il y a donc trois fichiers Logo enregistrés sur cette disquette. Lequel voulez-vous ramener en mémoire? Si c'est le premier,

RAMENE "MARDI13.LOG

Dès que le point d'interrogation réapparaît, tout le contenu du fichier se trouve dans l'espace de travail. Si les titres des procédures ramenées n'apparaissent pas sur l'écran, tapez IMTS pour les voir. Attention! à ce moment s'affichent tous les titres des procédures existant dans l'espace de travail (même celles qui existaient auparavant).

Eliminer un fichier.

Pour effacer un fichier de la disquette, tapez:

DETRUIS "MARDI

si le fichier à détruire s'appelle MARDI. Notez la différence entre la primitive EF (EFFACE) qui efface quelque chose de l'espace de travail, et la primitive DETRUIS qui efface un fichier de la surface de la disquette.

Ramenons un autre fichier en mémoire :

RAMENE "MARDI13

Pour ramener un fichier en mémoire

RAMENE "NOMFICHIER

Voyons ce qu'il y a maintenant dans l'espace de travail

IMTS
POUR PENTAGONE
POUR TOURNOI
POUR FLEUR
POUR TRIANGLE
POUR CERCLE

Séparons les procédures de ce fichier :

SAUVE "GEOM [PENTAGONE TRIANGLE CERCLE]

et ensuite :

SAUVE "MARDI [FLEUR TOURNOI]

Effaçons MARDI13 de la disquette :

DETRUIS "MARDI13

Voyons ce qu'il y a maintenant sur la disquette :

.SYSTEME (ENTREE)
DIR (ENTREE)

DERIVE
ETUDE
CHAP4
GEOM

MARDI
MARDI13 n'apparait plus, et les deux nouveaux fichiers GEOM et MARDI sont dans le catalogue.

Pour sauver une Image
Avec la même disquette dans le lecteur....

SAUVEIMAGE "DESSIN.DES

Chapitre 5

L'éditeur

Si vous voulez corriger une procédure, en définir une nouvelle ou écrire tranquillement du texte, vous utiliserez l'éditeur.

Quand vous êtes dans l'éditeur, le curseur peut se déplacer sur toute la surface de l'écran. Vous pouvez ainsi modifier, insérer ou supprimer des caractères, voire des lignes et des paragraphes entiers sans la moindre rature.

On peut comparer l'éditeur à un garage où il est possible de "réparer" des procédures. Vos outils : les touches de manipulation du curseur.

On peut aussi le comparer à un cahier. Dans celui-ci on écrit, on planifie, on définit ou on corrige, une série d'actions à faire exécuter par la suite, ou un texte qui n'est pas du Logo mais que vous voudrez enregistrer ou imprimer.

Souvenez-vous que tant que vous êtes dans l'éditeur, ce que vous tapez ne sera pas exécuté.

Entrer - Sortir de l'Editeur

On rentre dans l'éditeur en tapant

ED ou ED " nom de la procédure
suivi de la touche *(ENTREE)* bien-sûr. Une inscription, EDITEUR , ou la couleur de l'écran qui change vous rappellent dans quel monde vous vous trouvez. D'ailleurs, le symbole d'invite (?) n'est plus en début de ligne.

Vous pouvez maintenant taper ce que vous voulez.

Pour sortir de l'éditeur et retrouver le mode direct, deux possibilités :

a. **Les touches** *(CTL)* *(Q)*. On quitte l'éditeur en concluant l'édition. Ce que vous avez tapé entre dans l'espace de travail.

Voici la procédure que nous avions définie.

POUR CARRE
AVANCE 50 TD 90
AVANCE 50 TD 90
AVANCE 5 TD 90
AVANCE 50 TD 90
FIN

Et voici quel a été le résultat.

CARRE
Ce n'est pas vraiment un carré.

Amenons la procédure dans l'éditeur pour la corriger.

ED "CARRE *(ENTREE)*

Le nom de la procédure est précédé d'un guillemet (devinez ce qui se passerait si on l' oubliait). Et dans l'éditeur on retrouve :

POUR CARRE
AVANCE 50 TD 90
AVANCE 50 TD 90
AVANCE 5 TD 90. . . . . . . Cinq, au lieu de cinquante! Rajoutons un zéro! En appuyant sur les touches, on place le curseur juste après cinq, FIN et on retape 0. Le reste de la ligne se décale.
AVANCE 50 TD 90

Bon! Sortons d'ici.
Appuyons sur *(CTL)* *(Q)*

VOUS VENEZ DE DEFINIR CARRE

Tout ce qui existait avant de rentrer dans l'éditeur a disparu de l'écran . Le point d'interrogation est en début de ligne, allons-y!

CARRE
Ce qui est mieux!

b. **Les touches** *(CTL) (C)* . On arrête l'édition. Ce que vous avez tapé n'entre pas dans l'espace de travail mais reste dans la mémoire tampon de l'éditeur.

Manipulations dans le monde de l'éditeur.

Dans l'éditeur le curseur se déplace par des touches. Vous connaissez déjà certaines d'entre elles. En voici quelques autres.

(←)	Déplace le curseur vers la gauche.
(→)	Déplace le curseur vers la droite.
(↑)	Déplace le curseur vers le haut.
(↓)	Déplace le curseur vers le bas.
(<X)	Efface le caractère à gauche du curseur.
(<ESC>)	combiné à d'autres touches permet d'obtenir certaines commandes. Voir manuel de référence.

Pensez que quand vous appuyez sur la touche *(ENTREE)* elle déplace tout ce qui suit le curseur, sur la ligne inférieure, et c'est comme si vous aviez tapé un caractère. Ne vous étonnez donc pas si la ligne du dessous remonte en appuyant sur *(< X)* quand le curseur se trouve en début de ligne.

Définir des procédures dans l'éditeur.

On peut définir une procédure directement dans l'éditeur.

ED

Vous entrez dans l'éditeur qui est vide ou contient le texte de votre dernière édition. Tapez votre procédure : comme d'habitude, la première ligne est le titre (POUR....) , ensuite les lignes d'instructions, et finalement, le mot FIN seul sur la dernière ligne.

Pour définir plusieurs procédures les unes à la suite des autres, n'oubliez pas de mettre FIN à la fin de chaque procédure avant d'entreprendre la définition suivante qui doit commencer par POUR.

Dans l'éditeur, tapons une nouvelle procédure.

POUR FORME
TD 45 AVANCE 60
TD 90 AVANCE60 — Attention! si on place le curseur en début de cette ligne
TD 90 AVANCE 30........ — et appuyons sur (<X)
FIN

...l'action qui intervient entre AVANCE60 et TD 90 est l'appui de *(ENTREE)*. Cette action-ci est effacée par l'appui de *(< X)*.

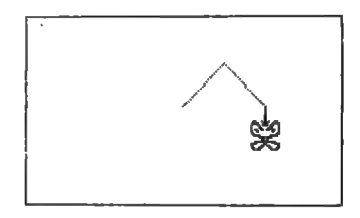
Voici donc ce qui arrive : la quatrième ligne remonte à la troisième.

POUR FORME
TD 45 AVANCE 60
TD 90 AVANCE60TD 45 AVANCE 30
FIN

Appuyons à nouveau sur *(ENTREE)* . Et nous retrouvons :

POUR FORME
TD 45 AVANCE 60
TD 90 AVANCE60. . . . . . . Il faut placer le curseur sur 6 et appuyer sur la barre
TD 45 AVANCE 30 — d'espacement pour insérer un espace.
FIN

Sortons de l'éditeur et demandons:



FORME
C'est ce qu'on voulait.

Faisons la même forme en tournant à gauche.

Nous remplacerons TD par TG. Editons d'abord la procédure :

ED "FORME

La voici :

POUR FORME
TD 45 AVANCE 60
TD 90 AVANCE 60
TD 45 AVANCE 30
FIN

Pour porter des modifications, rappelez-vous que le curseur se déplace n'importe où sur le texte de l'écran. Déplacez-le avec les touches décrites dans la section précédente.

Quand vous avez terminé les définitions, appuyez sur les touches *(CTL Q)* pour sortir de l'éditeur. Ce que vous avez fait entrera dans l'espace de travail et vous pourrez enfin faire exécuter des instructions.

Le message habituel apparaîtra: VOUS VENEZ DE DEFINIR.....

Utiliser l'éditeur pour corriger une procédure déjà définie.

Une procédure contient une erreur? Amenez-la dans l'éditeur pour la corriger. Vous taperez, par exemple :

ED "CARRE

le nom de la procédure est précédé de guillemets. Si vous voulez amener plusieurs procédures écrivez leur nom dans une liste:

ED [CARRE RECTANGLE]

Sur l'écran apparaît le texte des procédures demandées. Déplacez le curseur pour le corriger.

Modifier une procédure pour en faire une nouvelle.

D'une procédure en faire deux! Amenez une procédure dans l'éditeur.

ED "CARRE

et portez des modifications : changez la valeur des côtés, remplacez TD par TG, ou encore d'autres modifications.

Ensuite, changez le titre de la procédure. Vous pouvez rajouter une lettre au nom actuel (CARREG) ou le remplacer par un autre nom (BOITE au lieu de CARRE).
Quand vous sortirez de l'éditeur en appuyant sur *(CTL Q)*, vous aurez deux procédures dans l'espace de travail : CARRE (la première procédure) et BOITE ou CARREG (celle qui vient d'être créée).

Partout où il y a TD, il faudra taper TG.

Tapons <ESC> E . L'éditeur de texte vous demande quel est le mot à échanger. Tapez TD (ENTREE) , l'éditeur vous demande ensuite :
REMPLACER PAR : Tapez TG et appuyez sur *(ENTREE)*

POUR FORME
TG 45 AVANCE 60
TD 90 AVANCE 60
TD 45 AVANCE 30
FIN

Si vous voulez remplacer tous les TD de la procédure, il vous faudra recommencer autant de fois cette opération qu'il y a de TD.

Modifions le nom de cette procédure en rajoutant la lettre G à la fin de son titre (POUR FORMEG ... forme gauche)

En appuyant maintenant sur *< CTL Q >* on sort de l'éditeur et on obtient le message

VOUS VENEZ DE DEFINIR FORMEG

Nous venons de définir une nouvelle procédure FORMEG, et FORME existe toujours dans la mémoire sous son ancienne forme!...

Chapitre 6

De bonnes habitudes

Prenez de bonnes habitudes de programmation! Avec Logo, fini les longs programmes qui alignent des séries d'instructions difficiles à lire et impossibles à corriger. Une méthode s'impose : la programmation structurée.

La programmation structurée.

Pour aborder un "problème" inconnu, une bonne habitude consiste à le diviser en petites parties. Chaque partie est plus restreinte et peut être traitée séparément. Pour faire une figure complexe, par exemple, on peut faire tracer chaque forme qui compose cette figure par une procédure différente. On définira ensuite des procédures "générales" qui utiliseront comme "sous-procédures" celles qui ont déjà été définies.

Voici quelques avantages de cette façon de procéder:

-L'analyse du "problème global" en parties indépendantes et logiquement cohérentes, mène à la structuration du projet : premier pas vers sa solution.

-Chaque "partie" à résoudre (procédure à définir) est plus restreinte, sa réalisation est donc plus facile.

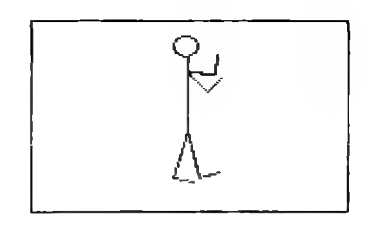
-Chaque unité est modulaire, elle peut être réutilisée dans un autre projet. Un carré, par exemple, peut représenter des objets différents dans différentes images.

-Le nom donné à chaque procédure peut exprimer ce qu'elle fait. Leur documentation se fait donc automatiquement. On sait qui fait quoi.

-La "chasse aux bugs*" devient plus facile. Pour trouver une erreur, ou modifier un point du programme, il n'y a pas besoin de lire toutes les instructions. On cherche directement dans la procédure "déficiente".

Cette méthode d'englobement de sous-procédures dans d'autres procédures, s'appelle **programmation structurée.**

* bug : punaise (en anglais) mais aussi erreur fourbe qui se cache dans un programme.



Regardez ces deux procédures :

```
POUR RICHARD
CT
TD 15 RE 50 TG 90 RE 20 AV 20 TD 90
AV 50 TG 30 RE 50 TG 90 RE 20 AV 20
TD 90 AV 50 TD 15 AV 60 TG 45 RE 30
TG 90 RE 20 AV 20 TD 90 AV 30 TG 40
RE 30 TG 90 RE 20 AV 20 TD 90 AV 30
TD 85 AV 15 TG 90
REPETE 36 [AV 2 TD 10]
FIN
```

```
POUR RICHARDII
CT
JAMBES
CORPS
BUSTE
TETE
FIN
```

RICHARD et RICHARDII dessinent exactement le même personnage.

RICHARD utilise uniquement des primitives, AV, RE, TG etc. C'est un très long programme, vous aurez du mal à le taper car vous risquez de vous perdre parmi toutes ces instructions.

RICHARDII utilise 5 mots, dont 4 procédures que nous avons créées. Regardons-les de plus près.

```
POUR JAMBES
TD 15 PIED
TG 30 PIED
TD 15
FIN
```

```
POUR CORPS
AV 60
FIN
```

```
POUR BUSTE
TG 45 BRAS
TG 40 BRAS TD 85
COU
FIN
```

```
POUR TETE
REPETE 36 [AV 2 TD 10]
FIN
```

Voyons le PIED

```
POUR PIED
RE 50 TG 90 RE 20
AV 20 TD 90 AV 50
FIN
```

Une idée : BRAS et PIED se ressemblent. Au lieu de retaper la procédure on a changé son nom et certains traits en utilisant l'éditeur.

Voyons le BRAS

```
POUR BRAS
RE 30 TG 90 RE 20
AV 20 TD 90 AV 30
FIN
```

et le COU

```
POUR COU
AV 15
TG 90
FIN
```

Procédures transparentes.

Faites attention à l'état de la tortue à la fin d'un dessin car c'est depuis cet état qu'elle repartira pour faire le prochain tracé. Si elle est mal positionnée ou mal orientée...tout ira de travers.

Une bonne habitude est donc de remettre la tortue à son état de départ à la fin d'une procédure. Les procédures transparentes sont celles qui laissent la tortue comme elle a démarré : elles vous évitent des surprises lors de l'enchaînement de deux procédures.

Attends un peu....

L'exécution d'une procédure se fait si rapidement! Il n'est pas toujours facile de suivre le parcours de la tortue pour voir où elle va. Une primitive vous sera utile pour ralentir ses déplacements : ATTENDS suivi d'un nombre fait attendre le déroulement d'une procédure pendant quelques dixièmes de seconde. Introduisez cette primitive à différents emplacements de la procédure.

ATTENDS, comme son nom l'indique, fera attendre le déroulement entre deux instructions et vous donnera le temps de suivre le parcours de la tortue. Quand vous aurez corrigé la procédure, enlevez les ATTENDS, le dessin se fera aussi vite qu'avant.

Le dessin animé.

Faire du dessin animé, serait-ce une bonne habitude? En tout cas, c'est une activité amusante.

Voici une idée pour faire bouger un dessin:

Tracez une figure, puis retracez-la en l'effaçant (par GC). Ensuite, déplacez un peu la tortue ou donnez lui un autre angle de départ, et recommencez. Si la tortue est cachée, le tracé se fera assez vite et vous aurez l'illusion du mouvement. Essayez! Vous comprendrez alors l'utilité des "bonnes habitudes" !...

Une autre idée : Utilisez la commande NETTOIE. Elle permet d'effacer les tracés de l'écran sans remettre la tortue à l'origine.

Nous vous montrons ici la forme finale des procédures que nous avons définies. Evidemment, il a fallu un certain tatonnement au départ pour arriver à faire converger l'analyse du problème avec la définition de chaque unité.

Et si RICHARDII bougeait ses jambes?

```
POUR EFJAMBES
GC   JAMBES BC
FIN
```

Pour faire marcher RICHARDII

```
POUR BOUGE
REPETE 5 [JAMBES EFJAMBES ]
JAMBES
FIN
```

Et enfin le dessin animé:

```
POUR ANIME
RICHARDII
ORIGINE   ........ cette primitive replace la tortue à l'origine (centre de l'écran, tête en l'air).
BOUGE
FIN
```

A vous de lui faire bouger les bras!
Essayez aussi de lui colorer les jambes.

Analyse d'un mouvement:

```
POUR LUMBAGO
RICHARDII  VE
RICHARDIII  VE
RICHARDIV VE
RICHARDV
FIN
```

La seule différence entre RICHARDII, RICHARDIII, RICHARDIV et RICHARDV est la procédure CORPS.

Nous avons défini CORPSIII, CORPSIV et CORPSV en ajoutant TD avec une donnée de plus en plus grande au début de la procédure.

Pouvez-vous faire le lumbago du roi?

Chapitre 7

Procédures avec donnée(s)

Pour faire des carrés de tailles différentes, faudra-t-il définir plusieurs procédures, une pour chaque taille? Après tout, un carré est toujours un carré, du moment qu'il a 4 côtés (peu importe leur taille) et 4 angles de 90 degrés.

En Logo, on écrit le minimum pour obtenir le maximum. On peut donc définir une procédure générale avec les actions de base du carré. Ce qui change à chaque fois, sa taille, sera précisé au moment de la demande d'exécution de la procédure.

Vous connaissez des primitives avec donnée (comme AVANCE 45) et des primitives sans donnée (comme LC). Eh bien, c'est pareil pour les procédures. Nous avons déjà vu des procédures sans donnée. Il suffit de taper simplement leur nom (comme TRAJET) sans l'accompagner de quelque chose. Il est temps de voir maintenant comment définir des procédures avec donnée.

Ecriture et demande d'exécution.

Voici la définition d' une procédure avec donnée*.

```
POUR CARRE :COTE
REPETE 4 [AVANCE :COTE TD 90]
FIN
```

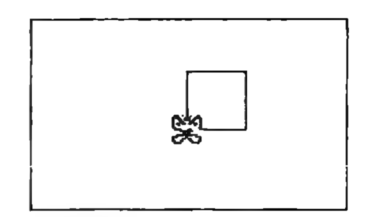
La donnée :COTE est annoncée dans le titre, après le nom de la procédure. Dans le corps de la procédure on retrouve cette donnée, là où on veut pouvoir faire varier. Attention aux deux points devant le nom de la donnée ; ils sont accolés au mot. Ces deux points signifient que l'ordinateur doit prendre en considération ce qui est désigné par ce mot et non le mot lui-même. Autrement dit, la donnée de CARRE est le nombre qui sera spécifié pour COTE et non pas le mot COTE**.

* d'autres termes utilisés sont : variable, argument, paramètre, entrée.

** Au lieu du mot COTE on aurait bien sûr pu écrire TAILLE, ou X ou C ou...mais il est plus pratique d'appeler les choses par ce qu'elles représentent.

Grands et petits carrés.

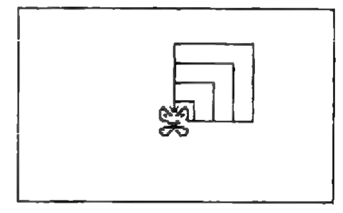
```
POUR CARRE :COTE
REPETE 4 [AV :COTE TD 90]
FIN
```



CARRE 60
CARRE
PAS ASSEZ DE DONNEES POUR CARRE

......même message que pour AVANCE quand on oublie d'indiquer le nombre....

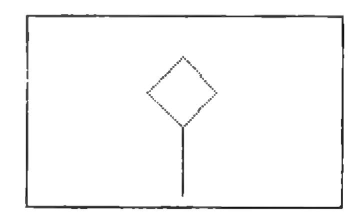
```
POUR IMAGE
CARRE 20
CARRE 40
CARRE 60
CARRE 80
FIN
```



IMAGE

POUR PANNEAU :X   n'oubliez pas d'indiquer la donnée dans le titre!!

```
TG 45
CARRE :X
TD 45
RE 70
FIN
```



PANNEAU 60

Et le rectangle.

```
POUR RECTANGLE :LONGUEUR :LARGEUR
REPETE 2 [AV :LONGUEUR TD 90 AV :LARGEUR TD 90]
FIN
```

RECTANGLE 60 20

Au moment de la demande d'exécution d'une procédure avec donnée, on précise la donnée par une valeur. Ainsi, on tapera CARRE 60, pour avoir un carré de 60 pas de côté, ou CARRE 85 pour un carré de 85 pas de côté.

Si la procédure a plusieurs données, elles doivent toutes apparaître dans son titre, précédées chacune de deux points, séparées entre elles par des espaces. Lors de la demande d'exécution de la procédure, à chaque nom de donnée est attribuée une valeur dans l'ordre de leur écriture.

Les données : des boîtes à remplir.

Pour expliquer le fonctionnement de procédures avec données, l'image d'une boîte que l'on remplit est souvent utilisée. L'ordinateur n'a évidemment pas de vraies boîtes dans sa mémoire, mais cette image aide à comprendre ce qui se passe.

En écrivant POUR CARRE :COTE, par exemple, on peut imaginer que l'on crée une boîte avec l'étiquette COTE dessus. Pour le moment elle est vide. Lorsqu'on demande l'exécution de CARRE 60, la valeur 60 rentre dans la boîte COTE. Pendant toute l'exécution de cette procédure, chaque fois que l'ordinateur rencontre :COTE il cherche ce qu'il y a dans la boîte. Dans notre cas, il trouve la valeur 60. C'est ainsi que la réalisation de CARRE 60 est un carré de 60 pas de coté.

Dès que l'exécution de cette procédure est terminée, la boîte se vide à nouveau. Cette caractéristique est importante à prendre en compte. Elle signifie que les valeurs données sont utilisées uniquement le temps d'exécution de la procédure dans laquelle elles interviennent. Par ailleurs, le nom choisi pour la "boîte", COTE , est propre à la procédure qui l'utilise. Dans une autre procédure le même nom de donnée peut être utilisé sans que ceci interfère avec la procédure CARRE. Il s'agit donc de variables locales.

Procédures imbriquées.

Les procédures avec donnée peuvent aussi être utilisées pour construire d'autres procédures.

Leur valeur peut être fixée (voir IMAGE). Si leur valeur n'est pas fixée, elles adoptent le nom des données de la procédure principale (voir PANNEAU).

Voici une autre procédure.

```
POUR SERIE :TAILLE
CARRE :TAILLE
DEPLACE :TAILLE
CARRE :TAILLE * 2
DEPLACE :TAILLE * 2
CARRE :TAILLE / 2
FIN
```

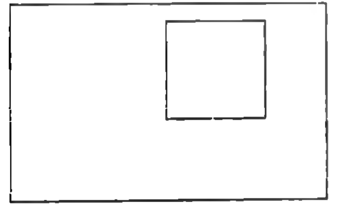
SERIE 50

A vous de définir DEPLACE :TAILLE pour pouvoir commander SERIE 50.
Pensez à utiliser LC pour faire déplacer la tortue sans laisser de traces.

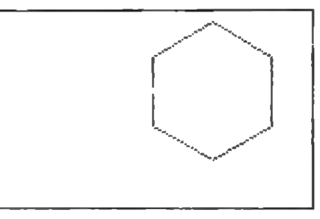
Tous les polygones en une seule procédure.

```
POUR POLYGONE :N :COTE
REPETE :N [AV :COTE TD 360 / :N]
FIN
```

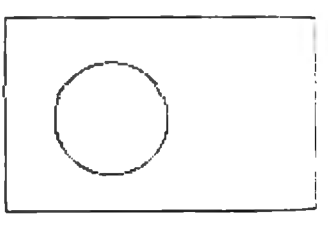
N représente le nombre de côtés du polygone et COTE, la taille de son côté.
Sur quelle loi de la géométrie tortue est basée cette procédure? Connaissez-vous le théorème du tour complet?
Essayez plusieurs valeurs pour POLYGONE.



POLYGONE 4 100



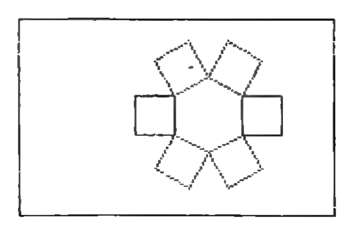
POLYGONE 6 80



POLYGONE 360 1

En combinant des polygones...

```
POUR FLEUR :TAILLE
POLYGONE 6 :TAILLE
REPETE 6 [AV :TAILLE TG 30 POLYGONE 4 :TAILLE TD 90]
FIN
```



FLEUR 50

Chaque fleur a sa taille...............

Chapitre 8

La récursivité

Regardons maintenant des techniques de programmation plus subtiles : les procédures récursives.

La récursivité est au coeur de la pensée informatique moderne. Nous présentons ici sa forme la plus simple : la récursivité terminale. Si cette problématique vous intéresse, vous découvrirez dans le chapitre 13 des formes plus sophistiquées.

Récursif mon cher Watson!

Imaginez qu'un jour vous rencontrez une fée. Une vraie fée, comme celle des contes, qui propose de réaliser deux de vos souhaits. Elle exécute le premier, et comme deuxième souhait vous lui demandez de réaliser deux de vos souhaits!

La récursivité, c'est ça : dans la définition d'une procédure, une des instructions est le nom de la procédure elle-même. Ainsi les actions de base sont décrites et par l'appel récursif elles se répètent à l'infini.

```
POUR CERCLE
AV 1 TD 1
CERCLE    appel récursif
FIN
```

Ceci est une procédure récursive. Si on demande à l'ordinateur d'exécuter CERCLE que va-t-il faire? Il commence par exécuter la première ligne, AV 1 TD 1, ensuite il passe à la deuxième qui est CERCLE, il fait donc AV 1 TD 1, et passe à la ligne suivante: CERCLE dont la définition est AV 1 TD 1 et CERCLE qui est..... heureusement qu'il y a des ordinateurs pour faire cela car nous aurions eu mal à la tête!

De plus en plus et de moins en moins.

Il est facile de décrire les mêmes actions sur des données légèrement différentes, dans la définition d'une procédure récursive. Il suffit de modifier les données de la procédure lors de l'appel récursif.

Des procédures qui se mordent la queue.
Appuyez sur *(CTL) (C)* pour les arrêter.

```
POUR BALADE
AVANCE HASARD 50
TD HASARD 360
FCC HASARD 8
BALADE
FIN
```

.... HASARD... voici une primitive intéressante. Elle reçoit un nombre comme donnée, et tire au hasard un nombre compris entre 0 et sa donnée - 1. HASARD 100 tire un nombre entre 0 et 99.

BALADE

Une autre façon de définir des polygones.

```
POUR POLY :COTE :ANGLE
AVANCE :COTE  TD :ANGLE
POLY :COTE :ANGLE
FIN
```

POLY 80 120 POLY 80 90 POLY 100 144

......pouvez-vous en faire d'autres?

POLY est inépuisable! Explorez ses possibilités.

Les carrés grandissent....

```
POUR CARRES :COTE
REPETE 4 [AV :COTE TD 90]
CARRES :COTE + 6
FIN
```

CARRES 3 ......appuyez sur *(CTL* C) pour arrêter.

En passant par l'éditeur on a rajouté un test d'arrêt: :

```
POUR CARRES :COTE
SI :COTE > 150 [STOP]
REPETE 4 [AV :COTE TD 90]
CARRES :COTE + 6
FIN
```

CARRES 3 ....et ça s'arrête tout seul!

Regardez la procédure CARRES. A chaque appel de la procédure, la tortue trace un carré. Mais, chaque carré aura des côtés qui font 6 pas de plus que le carré précédent.

Arrêt d'une procédure récursive.

Les procédures récursives, comme l'histoire de la fée, continuent à fonctionner encore et encore jusqu'à ce qu'elles soient arrêtées.

Vous pouvez arrêter manuellement une procédure en appuyant sur les touches *(CTL)* *(C)*.

Mais comme en Logo il s'agit de faire faire le travail par l'ordinateur vous pouvez introduire dans le texte de la procédure un **test d'arrêt.**

La deuxième version de CARRES contient un test d'arrêt.

SI :COTE > 150 [STOP]

sous-titrage français :

SI le côté est plus grand que 150, alors la procédure doit s'arrêter.

ou plus exactement:

SI la condition (:COTE > 150) est VRAI alors l'instruction contenue dans la liste (STOP) doit être exécutée.

SI reçoit 2 données: une condition et une liste d'actions à effectuer si la condition est remplie (c'est-à-dire son résultat est VRAI).

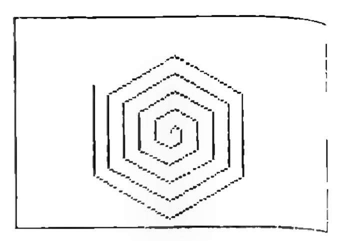
SI *condition* [*actions*]

Nous verrons plus loin que SI peut recevoir aussi 3 données.

SI ***condition*** [*actions si la condition est VRAI*] [*actions si la condition est FAUX*]

Mais, attention! Ecrivez **SI avec toutes ses données** sur la même ligne Logo (sans appuyer sur *(ENTREE)*).

Les spirales.



SPIRALE 2 60

```
POUR SPIRALE :COTE :ANGLE
AV :COTE  TD :ANGLE
SPIRALE :COTE + 3 :ANGLE
FIN
```

mettons un test d'arrêt: :

```
POUR SPIRALE :COTE :ANGLE
AV :COTE TD :ANGLE
SPIRALE :COTE + 3 :ANGLE
SI :COTE > 150 [STOP]
FIN
```

```
POUR SPIRALE :COTE :ANGLE
  SI :COTE > 150 [STOP]
AV :COTE TD :ANGLE
 SPIRALE :COTE + 3 :ANGLE
   FIN
```

Laquelle de ces deux spirales s'arrêtera toute seule? A vous de trouver.

Une spirale qui se déroule et s'enroule.

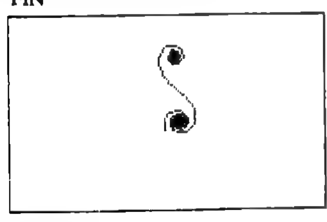
```
POUR SPIR :COTE :ANGLE
SI :COTE > 150 [GC SPIRI :COTE - 3 :ANGLE]
AV :COTE TD :ANGLE
SPIR :COTE + 3 :ANGLE
FIN
```

```
POUR SPIRI :COTE :ANGLE
SI :COTE < 2 [BC SPIR :COTE + 3 :ANGLE]
TG :ANGLE RE :COTE
SPIRI :COTE - 3 :ANGLE
FIN
```

Voyez-vous la double récursivité de ces deux procédures?

Des résultats surprenants!!!

```
POUR SPI :COTE :ANGLE
AV :COTE TD :ANGLE
SPI :COTE :ANGLE + 2
FIN
```



VE TG 90
SPI 5 15

VE TG 45
SPI 5 60

Voici un test d'arrêt pour la procédure POLY.

SI CAP = 0 [STOP]

CAP est une primitive qui informe sur l'orientation de la tortue. A l'origine, le cap de la tortue est 0 (elle pointe vers le haut de l'écran). Ce test d'arrêt dit : SI le cap de la tortue est égal à zéro (quand elle a son orientation d'origine) alors, STOP. La tortue s'arrêtera dès qu'elle aura retrouvé son orientation de départ. Introduisez ce test dans la procédure POLY!

Chapitre 9
Questions de syntaxe

En Logo, parfois il faut taper un guillemet devant un mot, parfois deux points, parfois rien du tout et d'autres fois des mots sont écrits entre crochets. Clarifions maintenant le sens de ces conventions.

L'exemple du langage parlé.

- TONGAI c'est le nom que j'ai donné à mon chien.
- Drôle de nom. Comment l'écris-tu?
- "TONGAI" c'est un T, un O, un N, puis G, A, et I.
- Connaît-il son nom?
- Bien sûr. TONGAI! Le voici...

Voici donc TONGAI écrit trois fois dans un dialogue, et à chaque fois l'utilisation du mot est différente.

TONGAI c'est le nom d'un chien.
"TONGAI" c'est le mot comme on peut l'épeler, une suite de lettres écrites dans un ordre précis (chaîne de caractères).
TONGAI! c'est l'ordre donné au chien de venir immédiatement près de son maître.

Un même mot peut donc avoir des usages différents dans le discours. Dans le langage parlé nous les comprenons par le contexte et l'intonation de la voix, et dans le langage écrit, par l'utilisation de symboles.

L'équivalent en Logo.

En Logo… c'est pareil. Un mot peut avoir différents sens.
Pour les distinguer, le mot doit être précédé par:
- Deux points pour signifier ce qui est désigné par ce mot,
:TONGAI c'est CHIEN
- Un guillemet pour signifier la suite de caractères du mot,
"TONGAI c'est T O N G A I (*)
- Rien du tout pour signifier l'action,
TONGAI c'est le nom d'une procédure.
-Des crochets pour délimiter une suite de mots, [TONGAI EST LA]

(*) Les nombres sont des mots particuliers en Logo. Ils ne sont pas nécessairement précédés par un guillemet.

TONGAI c'est...

...Un ordre

Voici sa définition :

```
POUR TONGAI
RECTANGLE 20 60 TD 180 PATTE  TG 90 AV 60...
TD 90 PATTE  TD 180 AV 20 TD 45 PATTE  TG 135
AV 60 RECTANGLE 20 20 CT
FIN
```

RECTANGLE est dans le chapitre 7.
Pouvez-vous définir PATTE?

Et son exécution :

TONGAI

...Un mot

```
EC "TONGAI
TONGAI
```

Ce mot a 6 éléments.

```
EC COMPTE "TONGAI
6
```

.....Le nom de quelque chose

```
DONNE "TONGAI  "CHIEN
```

```
EC :TONGAI
CHIEN
```

Cette "chose" a 5 éléments.

```
EC COMPTE  :TONGAI
5
```

Et si on tape :

```
DONNE "TONGAI [LE CHIEN DE SYLVAIN]
```

```
EC :TONGAI
LE CHIEN DE SYLVAIN
```

ce n'est plus CHIEN.

Ce que TONGAI désigne maintenant a 4 éléments.

```
EC COMPTE :TONGAI
4
```

C'est une liste a 4 mots.

Comment donner des noms à des choses.

Par l'écriture de procédures on nomme des actions.

Quand on écrit POUR TONGAI

....

....

FIN

TONGAI est défini comme une suite d'instructions. L'ordinateur les exécutera quand l'ordre TONGAI lui sera donné.

Voyons maintenant comment donner des noms à des choses. Ceci permet de stocker des informations en mémoire.
En Logo, une chose ou **objet Logo**, est tout ce qui n'est pas une action : un mot, un nombre ou une liste. Les objets Logo s'utilisent comme données de certaines primitives ou procédures.

Dans le chapitre 7 nous avons vu une façon de nommer des choses : leur nom apparaît dans le titre de la procédure.

Voici une autre façon de nommer des choses :

DONNE "TONGAI "CHIEN et :TONGAI est le mot CHIEN
DONNE "COTE 60 et :COTE est 60.

DONNE est une primitive avec deux données. La première est un mot : c'est le nom qu'on donne à la chose, la deuxième peut être un mot, un nombre, ou une liste.

Pour en revenir aux boîtes.

Rappelez-vous les boîtes du chapitre 7. DONNE est une primitive permettant de créer des boîtes qui sont remplies par un contenu fixe (la deuxième donnée de DONNE) et restent dans l'espace de travail* tant que l'on ne les efface pas. Même si ce contenu peut être momentanément remplacé par un autre, pendant l'exécution d'une procédure, elles retrouvent leur contenu initial dès que la procédure s'arrête de tourner.

Généralement, les noms créés avec DONNE sont des **variables globales**.

* Sauf si DONNE est utilisé dans le corps d'une procédure et que le nom de la chose apparait aussi dans le titre de la procédure .

Une chose peut contenir autre chose!..

DONNE "ANIMAL "MAMMIFERE
EC :ANIMAL
MAMMIFERE

DONNE :ANIMAL "CHIEN

EC :MAMMIFERE
CHIEN

Mettre des choses en boîte, ça peut toujours servir!

DONNE "COTE 100

Si on commande CARRE 80 ......voir la procédure du chapitre 7.
on aura un carré de 80 pas de côté.

Si on commande CARRE :COTE
savez vous combien de pas aura le côté du carré?

Pour mémoriser la position de la tortue.

Avec UNTRIANGLE la tortue revient à sa position de départ. Il suffit de donner un angle et le triangle se trace tout seul.

```
POUR UNTRIANGLE :ANGLE
DONNE "DEBUT POS
AV 50 TD :ANGLE
AV 70
FPOS :DEBUT
FIN
```

............POS rend les coordonnées de la position de la tortue sur l'écran à un moment donné. FPOS fixe la position de la tortue sur l'écran à un point précis. Voir POS et FPOS dans le manuel de référence.

UNTRIANGLE 40 UNTRIANGLE 120

Regardons ce qu'il y a dans les boîtes :
EC :DEBUT
0 0
EC :ANGLE
PAS DE CHOSE DONNEE A ANGLE

ANGLE est vide. Elle contient la valeur 40 (ou 120) uniquement tant que la procédure UNTRIANGLE est active.

Chapitre 10

L'arithmétique

Les ordinateurs calculent vite et bien! Tout le monde le sait. Regardons d'un peu plus près l'arithmétique Logo.

Utilisation des opérations.

Si on tape 3 + 7 l'ordinateur répondra QUE FAIRE DE 10. Il a calculé le résultat, mais proteste car il ne sait pas quoi en faire. Si on avait tapé ECRIS 3+7, il aurait écrit 10, si on demandait AVANCE 3+7 la tortue aurait avancé de 10 pas et si on voulait utiliser ce résultat à l'intérieur d'une procédure, l'ordinateur l'aurait fait. Mais il aurait fallu le lui dire !

A se rappeler : en Logo, il faut **toujours préciser à quoi on veut utiliser le résultat d'une opération.** Ce qui veut dire que **le résultat d'une opération est toujours la donnée d'une autre instruction.**

Les signes d'opérations sont les suivants:

- \+ pour l'addition
- \- pour la soustraction (attention! ce signe, s'il est précédé d'un espace et accolé au nombre, indique non pas l'opération de soustraction mais un nombre négatif)
- * pour la multiplication
- / pour la division

Les tests :

- = pour l'égalité entre deux nombres
- > si le nombre de gauche est plus grand que celui de droite
- < si le nombre de gauche est plus petit que celui de droite

Les nombres décimaux s'écrivent avec un point (6.78).

Ecriture des opérations.

Il y a deux façons d'écrire les opérations arithmétiques.

a. Le signe de l'opération est écrit entre les nombres : c'est l'écriture infixée (5/9).

Quelques calculs.

EC 24 + 87
111

EC SOMME 89 30
119

DONNE "NOMBRE 100

EC :NOMBRE - 30
70

TD 360 / :NOMBRE
La tortue tourne à droite de 36 degrés.

AVANCE SOMME 20 :NOMBRE
La tortue avance de 120 pas.

EC SOMME 2 3 6
5
QUE FAIRE DE 6

Pour faire additionner plus de deux nombres, additionnez-les par deux.
EC SOMME 2 SOMME 3 6
11

DONNE "TROIS 3
DONNE "DIX 10

98.5
ou
EC 3 * (40 - (8 + 30) /12)
110.5

Dans ce cas, les calculs se font dans l'ordre de priorité : d'abord les divisions, ensuite les multiplications, les soustractions et en dernier l'addition. Les tests d'égalité et d'inégalité sont effectués à la fin. Laissez un espace entre le signe de l'opération et les nombres.

b. Le signe de l'opération précède les nombres, c'est l'écriture **préfixée**. Les primitives SOMME, DIFF, PROD, DIV sont des opérations préfixées (DIV 5 9 donne le même résultat que 5 / 9). Les opérations préfixées s'effectuent dans l'ordre d'écriture.

L'informatique utilise surtout l'écriture préfixée : l'action est notée d'abord, ensuite l'objet sur lequel elle porte (sa donnée). Mais comme nous sommes habitués à la notation infixée, Logo permet d'utiliser les deux méthodes pour certaines opérations .

Lorsque vous écrivez des opérations, évitez de mélanger des écritures infixées et préfixées dans la même ligne Logo ; vous pourrez avoir des surprises.

La chaîne de production.

Chaque opération rend un résultat. Ce résultat est reçu comme donnée par une autre instruction qui l'utilisera pour une autre action. Ce principe est très important : c'est ce qui fait que la programmation en Logo n'est pas une simple juxtaposition d'actions mais une véritable construction.

Imaginons cet échange comme un travail à la chaîne. Ce que produit chaque primitive ou procédure (son résultat) est passé à la prochaine qui, elle, effectue aussi un certain travail sur le produit et le passe à la prochaine instruction qui, elle aussi, après avoir fait son travail le passe à la prochaine etc. On continue ainsi jusqu'au produit final qui vous est restitué pour être affiché, ou pour faire bouger la tortue, ou pour tout autre action que vous aurez commandée.

Des procédures qui produisent.

Comme les primitives-opérations que nous venons de voir, nous pouvons aussi définir des procédures-opérations. La primitive RENDS est alors utilisée dans la définition de la procédure. Cette primitive reçoit une donnée, et son action consiste à terminer la procédure en rendant son résultat à la primitive ou procédure qui l'a demandé (la procédure appelante). Cette dernière, reçoit le produit pour "travailler" dessus et le rend à la prochaine et la chaîne continue.....

```
EC PROD  2 DIFF  40 SOMME 8 DIV  30 12
59

      PROD  2                    29.5

                DIFF  40               10.5

                          SOMME 8           2.5

                                DIV   30   12
```

```
EC DIV  7665437446    0.09
8.517152E+0010
```

Ne vous étonnez pas si le nombre affiché contient la lettre E. Il s'agit de la notation exponentielle . les ordinateurs l'utilisent souvent pour écrire de trop grands ou de trop petits nombres.

Quelques procédures sur les nombres.

```
POUR AUCARRE :X
RENDS PROD :X :X
FIN

EC AUCARRE 4
16

AVANCE AUCARRE 10
```

La tortue avance de 100 pas

```
EC SOMME  25 AUCARRE 5
50
```

Comparez avec la procédure suivante.

```
POUR LECARRE  :X
EC PROD :X :X
FIN
```

Voyez-vous pourquoi les possibilités de cette procédure sont bien limitées?

La moyenne de deux nombres.

```
POUR MOYENNE :X :Y
RENDS  DIV  SOMME :X :Y  2
FIN

EC MOYENNE 40 60
50

EC MOYENNE AUCARRE 5 AUCARRE 2
14.5

EC AUCARRE MOYENNE 5 2
12.25
```

Chapitre 11

Et le texte !

Si vous pensez que Logo c'est la tortue, détrompez-vous. Logo c'est aussi la manipulation de listes et de mots. On oublie souvent cet aspect du langage qui permet de combiner des mots pour produire de nouveaux mots, construire des jeux de mots, réfléchir sur le langage...

L'ordre des mots.

L'ordinateur ne comprend pas la signification des mots comme nous. Il peut travailler sur leur forme car il considère les caractères d'un mot, ou les éléments d'une liste dans leur ordre d'écriture. Pour "produire du sens", nous utiliserons donc ce type de description.

Certaines primitives permettent de manipuler les mots et les listes. Elles servent à enlever un élément (PREM, SP, DER, SD), à construire des mots ou des listes en les soudant (MOT, LISTE, PHRASE ou PH), à les examiner (MEMBRE? VIDE? etc) ou à agrandir des listes (MP, MD) .

Note : les nombres sont aussi des mots en Logo et toutes les primitives décrites dans ce chapitre leur sont applicables.

À l'aide de ces primitives, vous définirez vous-mêmes d'autres fonctions dont vous aurez besoin pour vos projets (comme remplacer un mot par un autre dans une liste) .

Où on reparle d'opérations.

Rappelez-vous de l'image du travail à la chaîne du chapitre 10. Eh bien, ici c'est la même chose.

Comme les opérations arithmétiques, les primitives qui permettent de manipuler les mots et les listes sont aussi des opérations. Elles effectuent un travail sur leurs données (leur enlèvent un élément, les assemblent etc), et rendent le résultat à une autre instruction qui le reçoit comme sa donnée.

D'abord quelques primitives :

```
EC PREM "LIVRE
L

EC PREM [LIVRE]
LIVRE

EC DER "FETE
E

EC DER [A B C D E F]
F

EC MOT "A "G
AG

EC PHRASE "TA [RA MA]
TA RA MA

DONNE "LETTRES  [A B C]

EC   PHRASE  :LETTRES "D
A B C D

EC :LETTRES
A B C

EC SP :LETTRES
B C

EC NOMBRE? 3
VRAI

EC NOMBRE? [3]
FAUX

EC NOMBRE? PREM [3]
VRAI
```

```
EC SP "LIVRE
IVRE

EC SP [LIVRE]
[ ]

EC SD "FETE
FET

EC SD [A B C D E F]
A B C D E

EC   MOT  "PAR  "FUM
PARFUM

EC PHRASE  "PAR  "FUM
PAR  FUM

EC PHRASE "D  :LETTRES
D A B C

MONTRE LISTE MD  "D  :LETTRES
[ [A B C] D]
```

```
DONNE "NOMBRES [UN DEUX TROIS QUATRE CINQ SIX ]

EC ITEM 5  :NOMBRES
```

Si vous tapez :

PREM "MARMITE
QUE FAIRE DE M

L'ordinateur a bien pris le PREM caractère du mot MARMITE mais ne sait pas quoi en faire. Doit-il l'écrire, le combiner avec autre chose? Rien ne lui a été spécifié.

De la même façon, les procédures que vous définirez peuvent aussi être des opérations qui rendent un résultat. La primitive RENDS, utilisée à l'intérieur de la procédure, termine l'action de cette procédure, en rendant son produit à la primitive ou procédure qui l'a appelée.

Les procédures qui produisent un résultat sont des **procédures-opérations**.

Le mot vide et la liste vide.

Il existe un mot vide. C'est un mot sans caractères et on le note par un guillemet suivi d'un espace.

Il existe aussi une liste vide. Elle n'a aucun élément, et on la note par des crochets sans rien à l'intérieur.

VIDE? sert à examiner si un mot ou une liste sont vides. Elle rend la valeur VRAI si c'est le cas, et FAUX dans le cas contraire. Cette primitive est très utile pour définir des tests lors d'une procédure récursive portant sur les mots ou les listes.

Si par exemple on enlève un à un des éléments d'une liste ou d'un mot, il est imporant de savoir quand celui-ci est vide pour pouvoir s'arrêter.

SI VIDE? :DONNEE [RENDS [ ]]

Et quelques procédures.

Trouver la racine d'un verbe régulier:

```
POUR RACINE :VERBE
RENDS SD SD :VERBE
FIN

EC RACINE "MARCHER
MARCH
```

ou

```
POUR RACINE :VERBE
EC SD SD :VERBE
FIN

RACINE "PARLER
PARL
```

Conjuger un verbe au présent.

Remplissons des boîtes :

```
DONNE "PRON [JE TU IL NOUS VOUS ILS]
DONNE "TERM [E ES E ONS EZ ENT]
```

et ensuite la procédure :

```
POUR CONJUGUER :VERBE :PRON :TERM
SI VIDE? :PRON [STOP]
EC PHRASE PREM :PRON MOT RACINE :VERBE PREM :TERM
CONJUGUER :VERBE SP :PRON SP :TERM
FIN
```

laquelle des deux RACINE définies plus haut avons-nous utilisée?

```
CONJUGUER "CHANTER :PRON :TERM
JE CHANTE
TU CHANTES
IL CHANTE
NOUS CHANTONS
VOUS CHANTEZ
ILS CHANTENT
```

Pour construire des phrases aléatoires comme ...

L'ASSASSIN CONSULTE A MINUIT
LA VICTIME FRAPPE UNE POMME

Créons d'abord les boîtes :

```
DONNE "SUJET [ [L'ASSASSIN] [LA VICTIME] [LE JUGE] [LE MAIRE] ]

POUR PHRASES
RENDS PHRASE PHRASE ITEM 1 + HASARD 4 :SUJET  ITEM 1 + HASARD 4
      :VERBE ITEM 1 + HASARD 4 :COMPLEMENT
>FIN
```

. . . . . . .Note : cette ligne est un peu décalée pour indiquer qu'il s'agit de la même ligne Logo.

```
EC PHRASES
```

............imprédictible!

La récursivité dans les procédures-opérations.

La récursivité est souvent utilisée dans les procédures - opérations. Voici le raisonnement suivi habituellement dans la construction de telles procédures:

- Quelles sont les actions à effectuer si la donnée est vide (la condition).

- Quelles sont les actions à effectuer sur un élément de la donnée, et recommencer le traitement avec la donnée tronquée.

Le VRAI et le FAUX.

Certaines primitives rendent comme résultat les valeurs logiques VRAI ou FAUX. Ces primitives se terminent généralement par un point d'interrogation (EGAL? MEMBRE? VIDE? etc) et sont utilisées pour faire des tests. Ce sont des **opérateurs logiques.** Vous aurez peut-être besoin de construire les vôtres en fonction de vos projets. Dans ce cas, la procédure que vous définirez rendra le mot VRAI lorsque la condition nécessaire sera remplie.

Prenons un exemple : pour savoir si une liste contient un nombre vous construirez une procédure qui dira si un nombre (n'importe lequel) existe ou pas dans cette liste.

```
POUR UNNOMBRE? :LISTE
>SI VIDE? :LISTE [RENDS "FAUX]
>SI NOMBRE? PREM :LISTE [RENDS "VRAI] [RENDS
  UNNOMBRE SP :LISTE]
>FIN
```

UNNOMBRE? peut alors être utilisée dans:

```
POUR DETERMINER :DONNEE
>SI UNNOMBRE? :DONNEE [EC [IL Y A UN NOMBRE DANS
  LA LISTE] ] [EC [JE N'AI PAS TROUVE DE NOMBRE]
>FIN
```

Et si on fait:

```
DONNE "SUITE [A B 5 T G]

DETERMINER :SUITE
IL Y A UN NOMBRE DANS LA LISTE
```

Créer une petite banque de données.

```
POUR METTRE :DONNEE
>EXECUTE [DONNE "INFO MD :DONNEE :INFO]
>FIN

POUR AFFICHE :INFO
>SI VIDE? :INFO [STOP]
>TAPE ".....
>EC PREM :INFO
>AFFICHE SP :INFO
>FIN

POUR CONSULTER
>SI VIDE? :INFO [EC [AUCUNE INFORMATION]] [AFFICHE :INFO]
>FIN
```

La première fois il faut initialiser la boite,
DONNE "INFO [ ]

Ensuite...

```
METTRE [VOICI DES TRADUCTIONS UTILES]
METTRE [AVANCE FORWARD]

CONSULTER
....VOICI DES TRADUCTIONS UTILES
....AVANCE FORWARD
```

Vous pouvez continuer à remplir la "banque" et par la suite, construire des procédures permettant de trouver directement la traduction d'un mot précis.

Pour remplacer un mot dans une phrase.

```
POUR REMPLACE :LISTE :MOT1 :MOT2
>SI VIDE? :LISTE [RENDS [ ] ]
>SI EGAL? PREM :LISTE :MOT1 [RENDS PHRASE :MOT2 REMPLACE
        SP :LISTE :MOT1 :MOT2] [RENDS PHRASE PREM :LISTE
        REMPLACE SP :LISTE :MOT1 :MOT2]
>FIN

EC REMPLACE [IL FAIT BEAU] "BEAU "MOCHE
IL FAIT MOCHE
```

Pour éliminer un élément d'une liste.

```
POUR ELIMINE :ELEMENT :LISTE
>SI VIDE? :LISTE [EC PH :ELEMENT [N'EXISTE PAS DANS LA LISTE] RENDS :LISTE]
>SI EGAL? PREM :LISTE :ELEMENT [RENDS SP :LISTE]
>RENDS MPREMIER PREM :LISTE ELIMINE :ELEMENT SP :LISTE
>FIN

EC ELIMINE "LE [ LE LA LES]
LA LES
```

Chapitre 12

Dialogue avec un ordinateur

Souvent, un ordinateur pose des questions et, selon la réponse donnée, propose une chose ou une autre. Voyons maintenant comment construire des programmes interactifs pour dialoguer avec un ordinateur.

La construction de tels programmes est d'un grand intérêt pédagogique car il faut penser aux "connaissances" que doit avoir l'ordinateur pour dialoguer avec un utilisateur, bien poser les questions et prévoir les réponses qui peuvent être données. De cette façon l'apprentissage se fait tant sur le plan informatique que sur la maîtrise du sujet qu'il faut introduire. Car pour que l'ordinateur reconnaisse une "bonne réponse", il faut qu'elle lui ait été d'abord "enseignée".

Bonjour, comment vous appelez-vous?

Toutes les primitives LIS..... sont très utiles pour écrire des programmes interactifs. Elles arrêtent momentanément l'exécution d'un programme et lisent un message tapé au clavier (ou provenant d'un autre périphérique). L'ordinateur reçoit ce message, et fait dessus les traitements demandés, puis poursuit le déroulement du programme.

On peut demander le traitement du message pendant sa lecture, comme dans:

LIS PHRASE "BONJOUR LL

Mais ce qui est lu est oublié par la suite.

Pour stocker le message, il faut lui donner un nom (le mettre en boîte!)

DONNE "REPONSE LL

REPONSE contient ce qu'on a tapé au clavier. On pourra le retrouver à tout instant par :REPONSE et l'utiliser à différents moments du programme.

Tous les traitements sont permis. Ainsi l'exemple précédent devient:

EC PHRASE "BONJOUR :REPONSE

et cette même :REPONSE peut être réutilisée dans un autre traitement plus loin dans le programme quand par exemple on demandera :

SI MEMBRE? PREM :REPONSE :NOMS [EC [JE VOUS CONNAIS]]
[DONNE "NOMS MD PREM :REPONSE :NOMS]

Cette ligne Logo teste si le PREMier élément de :REPONSE (disons le nom d'une personne) est membre de :NOMS (disons une liste de noms que l'ordinateur a mémorisée). Si c'est le cas, la phrase JE VOUS CONNAIS est affichée ; si ce n'est pas le cas, la liste NOMS est agrandie en rajoutant le PREMier élément de :REPONSE comme dernier élément de cette liste.

Attention! LL rend toujours une liste. Même si vous tapez un seul caractère, il sera rendu à l'ordinateur sous forme de liste.

Pour lire un caractère.

Parfois une seule touche suffit pour répondre à un ordinateur. Certaines questions se contentent d'une simple réponse O (pour OUI) ou N (pour NON) ou une autre lettre dans le cas de questionnaires à choix multiples.

LISCAR est une primitive qui permet cette particularité. Elle arrête un déroulement et "lit" un caractère tapé. En appuyant sur une seule touche (celle qui est prévue dans le programme) l'ordinateur réagit immédiatement - même pas besoin d'appuyer sur *(ENTREE)*.

Chapitre 13

Tout est récursif!

Revenons à l'histoire de la fée du chapitre 8 et donnons en une autre version : le premier souhait fait à la fée est d'avoir encore deux souhaits. Et le premier souhait de cette nouvelle série est d'avoir encore deux souhaits, dont le premier souhait est de disposer de deux souhaits ...ce qui peut mener bien loin! Mais imaginez : dès qu'on arrête cette requête, il reste à réaliser tous les deuxièmes souhaits des demandes précédentes. Ainsi, de réalisation en réalisation, on remonte les niveaux jusqu'à arriver à la toute première requête.

La récursivité non-terminale.

Cette nouvelle version de l'histoire de la fée illustre bien comment fonctionne une procédure avec la récursivité non-terminale. Dans ces procédures, l'appel récursif ne se situe plus en dernière ligne comme dans le cas des procédures du chapitre 8 : il reste donc encore des actions à exécuter après l'appel récursif.

Grecque appelle grecque : un exemple.

Regardez l'exemple suivant.

```
POUR GRECQUE :X
SI :X < 5 [STOP]
AV :X TD 90
GRECQUE :X - 10
TG 90 RE :X
FIN
```

Après l'appel récursif (GRECQUE :X - 10) il y a encore les actions TG 90 RE :X.

Que se passe-t-il quand vous demandez l'exécution de GRECQUE 30?

Deux procédures récursives

```
POUR DECOMPTER :N
SI EGAL? :N 0 [STOP]
EC :N
DECOMPTER :N - 1
FIN
```

et

```
POUR RECOMPTER :N
SI EGAL? :N 0 [STOP]
RECOMPTER :N - 1
EC :N
FIN
```

Voici la différence quand on commande:

```
DECOMPTER 4
4
3
2
1
```

```
RECOMPTER 4
1
2
3
4
```

De même pour les mots ou les listes

```
POUR TRI :DONNEE
SI VIDE? :DONNEE [STOP]
EC :DONNEE
TRI SP :DONNEE
FIN
```

```
POUR TRIA :DONNEE
SI VIDE? :DONNEE [STOP]
TRIA SP :DONNEE
EC :DONNEE
FIN
```

Et si on commande,

```
TRI "CHER
CHER
CHE
CH
C
```

```
TRIA "CHER
C
CH
CHE
CHER
```

La récursivité dans le graphique

L'arbre.

```
POUR ARBRE :TAILLE
SI :TAILLE < 2 [STOP]
AV : TAILLE TG 45 ARBRE :TAILLE /2
TD 90 ARBRE :TAILLE / 2
TG 45 RE :TAILLE
FIN
```

ARBRE 50
Un arbre binaire.

Modifiez cet arbre en changeant la donnée des rotations.

On commence avec la première ligne d'instructions : le test.

30 n'est pas plus petit que 5, on passe donc à la deuxième ligne, AV 30 TD 90. Ensuite la troisième ligne est l'appel récursif : on rentre dans GRECQUE 20 (GRECQUE :X - 10) qui est un duplica de GRECQUE 30. Les actions sont identiques, elles portent simplement sur des données différentes.

Remarquez que la dernière ligne de la procédure (TG 90 RE 30) n'a pas pu être exécutée car on est passé à la procédure GRECQUE 20. Comme le deuxième souhait de notre histoire, cette ligne reste en attente pour pouvoir se réaliser.

On est passé donc à la procédure GRECQUE 20. Test d'arrêt, deux actions, et appel de la procédure GRECQUE 10. La dernière ligne reste aussi en réserve. GRECQUE 10 est actionnée, test d'arrêt, deux actions, appel de GRECQUE 0. GRECQUE 0 à son tour, recommence : test d'arrêt....STOP! **0 est plus petit que 5.** Le test est VRAI!!!

GRECQUE 0 ne peut plus continuer son action. Elle "rend la main" à GRECQUE 10 qui continue avec ce qui lui reste à faire : TG 90 RE 10. Ensuite il y a FIN et GRECQUE 10 rend le contrôle à GRECQUE 20 qui continue avec ce qui lui reste à faire : TG 90 RE 20, et passe le contrôle à GRECQUE 30 qui termine son travail et passe le contrôle...à vous! Oui, le point d'interrogation est là : l'ordinateur attend vos instructions.

Voici une schématisation de ce que l'on vient de décrire :

```
GRECQUE 30
SI 30 < 5 [STOP]
AV 30 TD 90
                               GRECQUE 20
TG 90 RE 30                    SI 20 < 5 [STOP]
FIN                            AV 20 TD 90
                                           GRECQUE 10
                               TG 90 RE 20 SI 10 < 5 [STOP]
                               FIN          AV 10 TD 90

                                                        GRECQUE 0
                                            TG 90 RE 10 SI 0< 5 [STOP]
                                            FIN
```

Des fractales

```
POUR VONKOCH :X
SI :X < 5 [AV :X STOP]
VONKOCH :X / 3 TG 60
VONKOCH :X /3 TD 120
VONKOCH :X / 3  TG  60
VONKOCH  :X  / 3
FIN

POUR FLOCON :X
REPETE 3 [VONKOCH :X TD 120]
FIN
```

FLOCON 60

Des triangle dans des triangles...

```
POUR TRIANGLES  :TAILLE
SI   :TAILLE < 2     [STOP]
AV  :TAILLE  TD  120  TRIANGLES  :TAILLE / 2
AV  :TAILLE  TD  120  TRIANGLES  :TAILLE / 2
AV  :TAILLE  TD  120  TRIANGLES  :TAILLE / 2
FIN
```

TRIANGLES   100

A vous de faire des carrés dans des carrés...

Et quelques procédures-opérations

La factorielle d'un nombre est le produit de ce nombre et tous les entiers inférieurs à ce nombre. La factorielle de 0 est égale à 1.

```
POUR FACTORIELLE  :N
SI :N < 0 [RENDS [PAS DE NOMBRE NEGATIF] ]
SI    :N =  0  [RENDS 1] [RENDS PROD  :N FACTORIELLE  :N - 1]
FIN

EC FACTORIELLE 5
120
```

Un mot à l'envers

```
POUR ENVERS :M
SI VIDE?  :M  [RENDS :M]
RENDS MOT DER :M  ENVERS  SD  :M
FIN

EC ENVERS "LOGO
OGOL
```

Chapitre 14
Outils utiles

Logo permet de construire vos propres outils. Les procédures présentées dans ce chapitre vous seront bien "outiles"! Pour le moment, recopiez celles qui vous intéressent pour les utiliser sans forcément chercher à tout comprendre. Au fur et à mesure que vous maîtriserez mieux le langage vous pourrez modifier ces procédures pour mieux les adapter à vos besoins.

Commenter....

La procédure suivante permet d'utiliser le reste d'une ligne Logo pour écrire des commentaires. L'ordinateur n'interprètera pas tout ce qui suit le point virgule.

```
POUR ; :REM
FIN
```

Un exemple d'utilisation: RPT applique une commande à toute une série d'objets Logo.

```
POUR RPT :CM :L  ;[L est une liste d'objets Logo -CM est une commande]
>SI VIDE? :L [STOP]    ;[si la liste est vide on arrête]
>EXECUTE LISTE :CM  MOT "" PREM :L
>RPT :CM SP :L  ;[appel récursif en tronquant la liste du premier élément]
>FIN
```

Des outils pour le graphique.

Pour ceux qui veulent définir un cercle à partir du rayon:

```
POUR CERCLE :RAYON
>DONNE "PI 3.1415926
>DONNE  "TAILLE  2 * :RAYON * :PI / 36
>REPETE 36 [TD 5 AV :TAILLE TD 5]
>FIN
```

De la même façon, vous pouvez définir un cercle en tournant à gauche.

La procédure suivante permet de définir un arc en précisant son rayon et le nombre de degrés :

```
POUR ARC :RAYON :DEGRES
>DONNE "TAILLE 2* :RAYON * :PI/36
>DONNE "LONG RESTE : DEGRES 10
>REPETE :DEGRES / 10 [TD 5 AV :TAILLE TD 5]
>SI :LONG > 0  [AV :TAILLE  * :LONG / 10 TD :LONG]
>FIN
```

Définissons la procédure RESTE.

```
POUR RESTE :DIVIDENDE :DIVISEUR
>RENDS :DIVIDENDE - ( :DIVISEUR * QUOT :DIVIDENDE :DIVISEUR)
>FIN
```

Des outils pour les mots et les listes.

POSITION rend la position dun caractère dans un mot ou d'un élément dans une liste (complémentaire à ITEM).

```
POUR POSITION :X  :LISTE
>SI NON MEMBRE? :X :LISTE [RENDS 0]
>SI :X = PREM  :LISTE [RENDS 1]
RENDS 1 + POSITION :X SP :LISTE
>FIN
```

```
ECRIS POSITION "C [A B C D E]
3
```

TRIER classe alphabétiquement les mots d'une liste.

```
POUR TRIER :LISTE
>RENDS CLASSER  :LISTE [ ]
>FIN
```

```
POUR CLASSER :LISTE :AUTRELISTE
>SI VIDE? :LISTE [RENDS :AUTRELISTE]
>RENDS CLASSER SP :LISTE INSERE PREM :LISTE :AUTRELISTE
>FIN
```

```
POUR INSERE :MOT :LISTE
>SI VIDE? :LISTE [RENDS MD :MOT [ ] ]
>SI ORDRE? :MOT PREM :LISTE [RENDS MP :MOT :LISTE] [RENDS MP
PREM :LISTE INSERE :MOT SP :LISTE]
>FIN
```

```
POUR ORDRE? :MOT1 :MOT2
>SI :MOT1 = " [RENDS "VRAI]
>SI :MOT2 = " [RENDS "FAUX]
>SI EGAL? ASCII :MOT1 ASCII :MOT2 [RENDS ORDRE? SP :MOT1 SP
:MOT2] [RENDS PLP? ASCII :MOT1 ASCII :MOT2]
>FIN
```

Ainsi on obtient:

```
ECRIS TRIER [DALI ARMEN SYLVAIN PATRÍCE]
ARMEN DALI PATRICE SYLVAIN
```

Ensembles et sous-ensembles.

Cette série de procédures permettent d'explorer les relations d'intersection ou d'union de deux ensembles, et d'examiner si un ensemble est un sous-ensemble d'un autre.

```
POUR EGALITE :A :B
>SI MOT? :A [RENDS :A = :B]
>SI MOT? :B [RENDS "FAUX]
>SI SOUSENS :A :B [RENDS SOUSENS :B :A]
>RENDS "FAUX
>FIN
```

```
POUR SOUSENS :A :B
>SI VIDE? :A [RENDS "VRAI]
>SI MEMBRE? PREM :A :B [RENDS SOUSENS SP :A :B]
>RENDS "FAUX
>FIN
```

```
POUR INTERSEC :A :B
>SI VIDE? :A [RENDS [ ] ]
>RENDS SI MEMBRE PREM :A :B [MP PREM :A INTERSEC SP :A :B]
    [INTERSEC SP :A :B]
>FIN
```

```
POUR UNION  :A  :B
>SI VIDE?  :A [RENDS :B]
>RENDS SI MEMBRE PREM :A  :B  [UNION SP :A
    :B] [MP PREM  :A UNION SP :A  :B]
>FIN

POUR MEMBRE  :A  :B
>SI VIDE?  :B [RENDS "FAUX]
>SI EGAL?  :A PREM :B [RENDS "VRAI]
>RENDS MEMBRE  :A  SP :B
>FIN

POUR MINUS :A :B
>SI VIDE?  :A [RENDS  [ ] ]
>RENDS SI MEMBRE PREM  :A :B [MINUS
   SP :A :B] [MP PREM :A MINUS SP :A  :B]
>FIN
```

MANUEL DE REFERENCE

La deuxième partie de cet ouvrage constitue le manuel de référence. Le manuel de référence a pour objet de répertortier et de décrire toutes les primitives présentes dans EXELOGO.

Afin de faciliter votre recherche nous avons essayé dans la mesure du possible de regrouper les primitives en rubriques distinctes selon leur domaine d'application. Ainsi, nous avons plusieurs rubriques : le graphique, les mots et les listes, les procédures, les noms, les valeurs logiques, les instructions conditionnelles et contrôles d'exécution, la gestion de l'espace de travail, l'éditeur, le monde extérieur et les fichiers. Par la suite, les primitives sont présentées par ordre alphabétique.

Si vous rencontrez des difficultés.

Si, malgré le manuel d'initiation et le manuel de référence, vous éprouvez des difficultés à maîtriser EXELOGO, nous vous rappelons que la revue **Exelement vôtre**, développe de nombreuses pages d'initiation aux divers langages de programmation. EXELOGO ayant été retenu par l'Education Nationale, la revue de l'EXL 100 consacrera plusieurs pages au langage LOGO.
De plus, il existe une grande variété d'ouvrage sur le langage LOGO. Pour mémoire, les éditions PSI, Edi Micros, Sybex, Cédic/Nathan, Hatier etc.. proposent dans leur catalogue de nombreux livres sur la célèbre tortue.

LE GRAPHIQUE

AVANCE,AV
BC
BC?
CAP
CC
CF
CT
ECH
ESTAMPE
FCAP
FCC
FCFG
FECH
FPOS
FX
FY
GC
LC
MT
NETTOIE
ORIGINE
PHOTO
POINT
POS
RECULE,RE
TD
TG
VE
VISIBLE?
XCOR
YCOR

Ces primitives permettent de faire du graphique en manipulant la tortue et son crayon.
L'**état** de la tortue est donné par sa **position** et son **orientation** dans le champ à un moment donné. Ses déplacements sont soit relatifs à son état, AVANCE , TD ..., soit absolus par rapport à des repères de l'écran (FCAP, FPOS).
Le **crayon** de la tortue est caractérisé par : sa **situation** (LC, BC, GC) et sa **couleur** . Celle-ci est représentée par un code numérique.
Certaines commandes qui "fixent" une action, ont leur contrepartie : une opération qui rend l'information correspondante (FCAP et CAP etc..).
Dans le chapitre sur les LUTINS vous trouverez des primitives qui permettent de manipuler les tortues-lutins.

LES LUTINS

CFORME	FORME
CHACUN	FVIT
CHOC?	FVMARCHE
COPIEFORME	GELE
DEGELE	PARLE
DEMANDE	QUI
EDCAR	SURCOULEUR
EDFORME	TAPISSE
FCFORME	VIT
FFORME	VMARCHE

EXEL-LOGO possède 4 tortues au lieu d'une seule. Ces tortues seront appelées lutins (SPRITES). A tout moment vous pourrez modifier leurs formes, leurs vitesses, leurs positions, leurs couleurs. Chaque tortue est repérée par un code, allant de 0 à 3. Chaque forme que vous définirez est aussi repérée par un code (de 0 à 15). Attention, les codes pour la forme et les codes pour la tortue sont indépendants, c'est à dire que, à la tortue 0 vous pouvez donner n'importe quelle forme, allant de 0 à 15.

La tortue active est celle à laquelle on s'est adressé en dernier par la commande PARLE.

Dans ce chapitre vous trouverez les primitives relatives à la manipulation des lutins. Les primitives SAUVEIMAGE, SAUVEFORME , SAUVECAR ainsi que RAMENEIMAGE, RAMENEFORME et RAMENECAR se trouvent dans le chapitre des fichiers.

LES NOMBRES
LES FONCTIONS MATHEMATIQUES

ACOS
ARRONDIS
ASIN
ATN
COS
DIFF (-)
DIV (/)
ENTIER
HASARD
PLG? (>)
PLP? (<)
PROD (*)
QUOT
RC
SIN
SOMME
TAN

Les nombres

En LOGO, vous pouvez utiliser les nombres entiers et les nombres décimaux positifs ou négatifs: par exemple 5 et - 10 sont des entiers, -3.05 et 1.5 sont des nombres décimaux.

Les nombres sont des mots particuliers. Vous pouvez ne pas les faire précéder de guillemet. Toutes les primitives des mots peuvent leur être appliquées. Par exemple:

```
?EC MOT? 425
VRAI
?EC COMPTE 425
3
```

Dans le dernier cas, LOGO rend le nombre de caractères du mot 425, c'est à dire 3 chiffres.

Un nombre contient au moins un chiffre et il ne peut y avoir d'espaces dans un nombre car les espaces sont des séparateurs. Pour la même raison, le signe moins (-) doit être accolé au chiffre pour indiquer un nombre négatif.

Les notations

Les nombres peuvent être aussi écrits en notation exponentielle (ou notation scientifique). Celle-ci consiste en un nombre multiplié par une puissance de 10. Par exemple:

7.2×10^3 représente 7.2 fois 1000 (10 puissance 3) ou autrement dit, 7200. Ce format se réfère à la notation scientifique qui est très utile pour comparer entre eux des nombres très grands ou très petits. Par exemple:

[illegible] $\times 10^{-7}$ et 2.03×10^{-4} sont tous les deux des nombres très petits, et cette notation permet de voir immédiatement que le premier nombre est près de 1000 fois plus petit que le second, cf les puissances de 10 diffèrent de 3 unités.

Les ordinateurs peuvent reconnaître et convertir des nombres en notation scientifique. Au lieu d'utiliser le chiffre 10 pour désigner la partie "puissance", ils utilisent un nombre suivi de la lettre E et un entier.

Un ordinateur utilise la deuxième écriture pour représenter ce nombre:

360×10^{-3} et 360E\-3 *

La notation scientifique stricte qui est celle que l'ordinateur rend toujours, le nombre qui précède la lettre E est plus grand ou égal à 1 et inférieur à 10. Voici la différence entre les notations scientifique et exponentielle.

Exponentielle	Scientifique
1E2	1.0E2
21.63E-4	2.163E\-3
0.4E0007	4.0E6

Notez cependant que quelque soit son écriture, seule la valeur du nombre est considérée.

TEC 00000
0

* *Attention, les nombres suivis du signe - (moins) ou du signe + (plus) tels que 1E-10 ou 1E+10 nécessitent le symbole \ pour annuler l'effet des opérateurs + et - qui font office de séparateur.*
Exemple:
1E-10 s'écrira 1E\-10

?EC COMPTE 00000
1
?EC 1E2
100

?EC SP 1E2
00

Les opérations

LOGO fournit des primitives qui permettent d'effectuer les opérations arithmétiques, la racine carrée d'un nombre etc...

Dans le cas des opérations arithmétiques, il existe une forme préfixée et une forme infixée. Par exemple :

?SOMME 3 15 et 3+15

rendent toutes deux 18, mais la première est la forme préfixée (l'opérateur précède les deux nombres) et la deuxième est la forme infixée (les deux nombres entourent le signe de l'opération).

LES MOTS ET LES LISTES

ASCII	LISTE	NOMBRE?
CAR	LISTE?	PHRASE, PH
COMPTE	MD	PREM
DER	MEMBRE?	SD
EGAL? (=)	MOT	SP
ITEM	MOT?	VIDE?
	MP	

Ces primitives servent à examiner, à démonter ou à construire des objets Logo, c'est-à-dire des mots et des listes. Ces primitives sont des opérations qui portent sur des objets Logo et leur résultat est aussi un objet Logo.

Un **mot** est composé de caractères : lettres, chiffres, signes de ponctuation. Chaque caractère est un élément du mot. La barre inversée (\) permet d'écrire des mots comprenant un caractère séparateur.

ECRIS "SAINT\ MICHEL

SAINT MICHEL

Les nombres sont des mots spéciaux en Logo, ils n'ont pas besoin d'être précédés de guillemet.

Une **liste** est une suite ordonnée d'objets Logo (mots ou autres listes) qui en sont les éléments.

Le tableau suivant compare les cinq primitives qui construisent de nouveaux objets Logo à partir de leurs données.

primitive	donnée1	donnée2	résultat
MOT	"BON	"JOUR	BONJOUR
MOT	"BON	[JOUR]	erreur
LISTE	"BON	"JOUR	[BON JOUR]
LISTE	"BON	[JOUR]	[BON [JOUR]]
LISTE	[BON]	"JOUR	[[BON] JOUR]
LISTE	[BON]	[JOUR]	[[BON] [JOUR]]
PHRASE	"BON	"JOUR	[BON JOUR]
PHRASE	"BON	[JOUR]	[BON JOUR]
PHRASE	[BON]	"JOUR	[BON JOUR]
PHRASE	[BON]	[JOUR]	[BON JOUR]
MP	"BON	"JOUR	erreur
MP	"BON	[JOUR]	[BON JOUR]
MP	[BON]	[JOUR]	[[BON] JOUR]
MD	"BON	"JOUR	erreur
MD	"BON	[JOUR]	[JOUR BON]
MD	[BON]	[JOUR]	[JOUR [BON]]

LES PROCEDURES

FIN
POUR
PRIM
PRIM?
PROC?

Une procédure est un programme Logo. De nouveaux mots sont définis à partir du vocabulaire connu du langage. Une procédure définie, est employée de la même façon qu'une primitive: elle peut être exécutée ou utilisée dans la définition d'une autre procédure.

Voir aussi le chapitre sur l'éditeur.

LES VALEURS LOGIQUES

ET
NON
OU

Ces primitives permettent d'effectuer les opérations logiques traditionnelles. Leurs données sont des prédicats c'est-à-dire des fonctions dont le résultat est le mot VRAI ou le mot FAUX.

LES NOMS

CHOSE
DONNE
NOM?

Ces primitives concernent les noms et les objets Logo qu'ils désignent.

Un nom est un mot qui sert à désigner un objet Logo (nombre, mot ou liste). Cet objet Logo est appelé la "chose" du mot. En Logo, cette "chose", si c'est un mot, peut être le nom d'une autre "chose".

INSTRUCTIONS CONDITIONNELLES ET CONTROLE D'EXECUTION

ATTENDS
EXECUTE
FAUX
LOGO
RENDS
REPETE
SI
STOP
VRAI

Ces primitives qui permettent de contrôler l'ordre dans lequel Logo suit vos instructions. Il s'agit :

- des instructions conditionnelles (si telle condition est réalisée faire telle suite d'instructions, sinon faire telle autre).

- des instructions d'itération (répétition de l'exécution d'une liste d'instructions).

- des instructions d'arrêt et d'interruption d'une procédure.

- des instructions d'exécution d'une liste d'instructions.

GESTION DE L'ESPACE DE TRAVAIL

CONTENU	IM
EF	IMNS
EFN	IMPS
EFNS	IMTOUT
EFPS	IMTS
.EFTOUT	

Ces primitives vous permettent de savoir ce que contient votre espace de travail, ou de le modifier.
L'espace de travail est une zone de la mémoire centrale qui contient les procédures et tous les noms de données que vous avez créés. Les primitives ne sont pas comprises dans l'espace de travail.

L'EDITEUR de TEXTE

L'éditeur de texte est un outil très pratique lorsque vous êtes dans le mode éditeur. Faites attention à ne pas confondre le mode éditeur qui est accessible par la primitive ED et le mode éditeur de texte qui est un autre niveau de l'éditeur.
Le passage dans le mode éditeur de texte se fait par pression de la touche <ESC>ape.
L'abandon d'une commande se fait également par cette touche (sauf recherche et échange de mots) .
Les commandes de l'éditeur de texte permettent de se déplacer dans le texte, d'insérer des caractères, supprimer des données, substituer des mots, copier et mouvementer des blocs.

DEPLACEMENT

<-	*déplacement vers la gauche*
->	*déplacement vers la droite*
flèche bas	*déplacement vers le bas*
flèche haut	*déplacement vers le haut*
touche HOME	*positionnement en début d'écran*

FONCTIONS PARTICULIERES

<ESC>

D(ébut)
- **E (écran)**
- **L (ligne)**
- **T (texte)**

F(in)
- **E (écran)**
- **L (ligne)**
- **T (texte)**

P(age)

- **P(récédente)**
- **S(uivante)**

M(ot)

- **P(récédent)**
- **S(uivant)**

S(uppression)

- **D(ébut)**
  - **L(igne)**
  - **T(exte)**
- **F(in)**
  - **L(igne)**
  - **T(exte)**

- **B(loc)**
- **M(ot)**
  - **P(récédent)**

  - **S(uivant)**

- **C(opie bloc)**
- **M(ouvemente bloc)**
- **D(marque début bloc)**
- **F(marque fin bloc)**
- **Q(uitte éditeur)**
- **E(échange mot)**
- **T(rouve mot)**

LE MONDE EXTERIEUR

- BOUTON?
- CFT
- CTX
- CURS
- ECRIS,EC
- ENTREE
- FCFT
- FCURS
- FCT
- .FSERIE
- LISCAR
- LL
- MANETTE
- ME
- MONTRE
- .SERIE
- SORTIE
- TAPE
- TE
- TOUCHE?

Ces primitives permettent de gérer des périphériques tels que l'écran, le clavier ou des manettes de jeu.
Au démarrage de Logo, le texte sur l'écran peut s'étendre sur 40 colonnes et 25 lignes.

LES FICHIERS

AVEC
.BCHARGE
.BSAUVE
CHARGE
.DEPOSE
.EXAMINE
DETRUIS
FLECTEUR
LECTEUR
RAMENE
RAMENECAR
RAMENEIMAGE
RAMENEFORME
SAUVE
SAUVECAR
SAUVED
SAUVEFORME
SAUVEIMAGE
.SYSTEME

Vous pouvez conserver votre travail sur cassette, Exelmemoire ou disquette. L'information y est alors organisée en fichiers. C'est dans ces fichiers que vous sauvegardez vos procédures ou vos données. Vous les réutiliserez ultérieurement sans avoir à les retaper au clavier, sans avoir à les redéfinir.
Les fichiers portent un nom dont le nombre de caractères maximum depend du peripherique (13 pour la cassette), suivi d'un suffixe facultatif formé d'un point et de 3 lettres. Un nom de fichier est un mot Logo qui ne comporte ni parenthèses, ni deux points (:), ni accents, ni point (sauf pour le suffixe).
Les primitives décrites dans ce chapitre nous permettent de travailler avec les fichiers.

ACOS n   opération

(ARC COSINUS) Rend l'arc cosinus de n en degrés.

Exemple:

?EC ACOS 0.707106
45.0000

ARRONDIS n — opération

Rend n arrondi à l'entier le plus proche. Comparez avec ENTIER.

Exemple:

```
?EC ARRONDIS .5
1
?EC ARRONDIS -5.38
-5
?EC ARRONDIS (6.3 + 8.4)
15
?EC ARRONDIS ((ARRONDIS 6.3)+8.4)
14
```

ASIN n — opération

(ARC SINUS) . Rend l'arc sinus de n en degrés.

Exemple :

?EC ASIN 0.707106
44.9999367

ATN y — opération

(ARC TANGENTE). Rend l'arc en degrés dont la tangente est y.

Exemple:

```
?EC ATN 3
71.56505118
```

ATTENDS n   commande

Fait attendre une exécution pendant environ n 60èmes de seconde. La donné de ATTENDS est arrondie à l'entier le plus proche.

Exemple
La procédure suivante fait des multiplications aléatoires. ATTENDS sert à donner le temps à l'utilisateur de lire le résultat.

```
POUR MULT
>DONNE "RESULTAT PROD HASARD 10 HASARD 20
>ECRIS :RESULTAT
>ATTENDS 45
>MULT
>FIN

?MULT
99
72
56
```

AVANCE n commande
AV n

Fait avancer la tortue dans la direction en cours de n pas.

Exemple:

?VE
? AVANCE 200

AVEC <u>module</u> commande

Installe des modules extérieurs contenant des primitives spéciales ou des extensions de l'espace de travail LOGO

ASCII mot — opération

Rend le code ASCII du PREM caractère de mot. Si mot contient un seul caractère rend le code ASCII de ce caractère. Voir aussi CAR .

Exemple :

```
?ECRIS ASCII  "A
65
?ECRIS ASCII  "BARBE
66
```

Les procédures suivantes permettent de définir un code secret :

```
?POUR ENCODER :LETTRE
>DONNE  "NUM (3 + ASCII  :LETTRE)
>SI :NUM > ASCII  "Z  [DONNE "NUM :NUM  -  26 ]
>RENDS CAR :NUM
>FIN

?POUR CODESECRET :MOT
>SI VIDE? :MOT [RENDS "  ]
>RENDS MOT ENCODER PREMIER :MOT CODESECRET SP :MOT
>FIN

?ECRIS CODESECRET  "EXELVISION
```

BC commande

Baisse le crayon de la tortue. Lors de ses déplacements elle laissera un tracé dans la couleur du crayon en cours. Après l'exécution de LC (LèveCrayon) ou GC (GommeCrayon) , la commande BC permet de continuer à tracer.

Exemple :

```
?AVANCE 40
?LC
?AVANCE 40
?BC
?AVANCE 40
```

BC? opération

Cette opération rend l'état du crayon. Si le crayon est levé, l'opération BC? rendra FAUX ; si le crayon est baissé , l'opération BC? rendra VRAI.

Exemple:

? BC
? EC BC?
VRAI

.BCHARGE nomfichier n1 n2 commande

Cette primitive permet de charger un programme assembleur avec les paramètres nomfichier n1 n2

n1	SEGMENT MEMOIRE 0 VDP 1 CRAM 2 RAM DISQUETTE 3 Non autorisé.
n2	ADRESSE Adresse de chargement
nomfichier	nom de fichier

## BOUTON? n		opération

Rend VRAI si le bouton de la manette n (n = 0 pour la manette orange et n = 1 pour la manette blanche) est pressé, sinon rend FAUX. Si les appareils ne sont pas branchés, BOUTON? rendra FAUX. Voir aussi MANETTE.

Exemple :

La procédure suivante permet de conduire la tortue avec les boutons de la manette. Si le bouton dela manette 0 est pressé, elle tournera à droite, si celui de la manette 1 est pressé, elle tournera à gauche, sinon elle ira tout droit.

```
?POUR CONDUITE
>SI LISCAR = "F [STOP]
>SI BOUTON? 0 [TD 15]
>SI BOUTON? 1 [TG 15]
>AVANCE 10
>CONDUITE
>FIN
```

Pour arrêter, appuyer sur F.

.BSAUVE	nomfichier n1 n2	commande

Cette primitive permet de sauvegarder un programme assembleur. Les paramètres à passer dans la commande .BSAUVE suivent les mêmes règles que .BCHARGE.

n1	SEGMENT MEMOIRE 0 VDP 1 CRAM 2 RAM DISQUETTE 3 Non autorisé.
n2	ADRESSE Adresse de chargement
nomfichier	nom de fichier

CAP opération

Rend le cap actuel de la tortue : un nombre décimal compris entre 0 et 359,999. Le cap est repéré comme sur une boussole : le Nord (0) est en haut de l'écran, l'Est (90) est à droite, le Sud (180) est en bas, et l'Ouest (270) à gauche. Au démarrage, le cap de la tortue est zéro. Voir aussi FCAP.

Exemple :

```
?FCAP 90
?EC CAP
90
```

CAR n — opération

(Caractère). Rend le caractère qui correspond au code ASCII de n. n doit être un entier compris entre 0 et 255. Voir aussi ASCII .

Exemple :

```
?ECRIS CAR 65
A
```

La procédure MAJUSCULES rend sa donnée écrite en lettres majuscules.

```
?POUR MAJUSCULE :MOT
>SI VIDE? :MOT [RENDS " ]
>RENDS MOT MAJ PREMIER :MOT MAJUSCULE SP :MOT
>FIN

?POUR MAJ :MOT
>SI (ASCII :MOT) < 96 [RENDS :MOT]
>SI (ASCII :MOT) > 122 [RENDS :MOT]
>RENDS CAR (ASCII :MOT) - 32
>FIN

?ECRIS MAJUSCULE "armen
ARMEN
```

CC opération

(Couleur Crayon) Rend un nombre représentant le code de la couleur actuelle du crayon. Par défaut, la couleur du crayon est le bleu (4).

code	couleur
0	Noir
1	Rouge
2	Vert
3	Jaune
4	Bleu
5	Magenta
6	Cyan
7	Blanc

Exemple :

```
?FCC 2
?AVANCE 100
?EC CC
2
```

CF opération

(Couleur Fond) Rend un nombre représentant le code de la couleur du fond graphique. Au démarrage, la couleur du fond est le rouge (1). Les codes des couleurs pour le fond sont les mêmes que celles pour le crayon: voir CC et FCFG.

Exemple:

```
?FCFG 4
? EC CF
4
```

CFORME opération

(Couleur FORME). Rend un nombre représentant le code couleur de la tortue active. Si vous avez activé plusieurs tortues en même temps en faisant par exemple PARLE [0 1 2 3], CFORME rendra la couleur de la première tortue de la liste, c'est-à-dire la couleur de la tortue 0. Voir FCFORME .

Exemple:

```
?FCFORME 2
? EC CFORME
2
```

CHARGE nomfichier commande

Transfère dans l'éditeur sans l'interpréter le contenu du fichier nomfichier . Pour interpréter ce qui a été chargé il faut entrer dans l'éditeur et taper <CTL Q>. Attention si le fichier est trop grand, il risque de ne pas tenir dans l'éditeur.

CHACUN <u>liste</u> commande

Fait exécuter les instructions de <u>liste</u> par chacune des tortues activées par la commande PARLE.

Exemple:

? VE
?AV 20
?PARLE [0 1]
?MT
? CHACUN [REPETE 4 [AV 20 TD 45 AV 20 TD 45]]

CHOC? n1 n2 — opération

Rend VRAI si la tortue n1 entrechoque la tortue n2.

Exemple:

```
?VE
?AV 20
?PARLE 1
?MT
? EC CHOC? 0 1
FAUX
```

CHOSE nom   opération

Rend l'objet Logo désigné par nom. nom à été créé comme nom de donnée par l'utilisation de DONNE ou en définissant une procédure.

CHOSE "TOTO est équivalent à :TOTO.

Exemple :

```
?DONNE "LETTRES [ A B C]
?ECRIS CHOSE "LETTRES
A B C
?DONNE PREM CHOSE "LETTRES   "a
?ECRIS :A
a
```

La procédure suivante affiche l'objet Logo de chaque nom dans une liste.

```
?POUR ECRISOBJ :L
>SI VIDE? :L [STOP]
>ECRIS CHOSE PREM :L
>ECRISOBJ SP :L
>FIN
```

Si nous faisons :

```
?DONNE "ARBRE "OLIVIER
?DONNE "OLIVIER [HUILE D'OLIVE]

?ECRISOBJ   [ARBRE OLIVIER]
OLIVIER
HUILE D'OLIVE
```

COMPTE obj   opération

Rend le nombre d'éléments de obj, obj étant un mot ou une liste.

Exemple :

```
?ECRIS COMPTE [A B C D E F]
6
?ECRIS COMPTE [ [A B] C D [E F] ]
4
?ECRIS COMPTE "BONJOUR
7
?ECRIS COMPTE 34
2
?DONNE "DALTON [ GERARD GEORGES GILBERT  GASTON ]
?ECRIS COMPTE :DALTON
4
```

Nous vous donnons trois procédures. Les deux premières permettent de remplacer certaines primitives qui n'existent pas dans EXELOGO.

```
?POUR DONLIS :MOT
>SI VIDE? :MOT [RENDS :LISTE]
>DONNE "LISTE PH :LISTE PREM :MOT
>DONNE "MOT SP :MOT
>FIN

?POUR CRELIS :MOT
>DONNE "LISTE [ ]
>DONLIS :MOT
>FIN
```

La procédure suivante permet de choisir au hasard un élément d'un mot ou d'une liste.

```
?POUR COURTEPAILLE :DONNEE
>ECRIS ITEM (1 + HASARD COMPTE :DONNEE) CRELIS :DONNEE
>FIN
?COURTEPAILLE :DALTON
GASTON
```

CONTENU opération

Rend une liste avec tous les mots crées par l'utilisateur.

Exemple:

?DONNE "TOTO "TATA
?CONTENU
TOTO
TATA

COPIEFORME n1 n2 commande

Copie la forme n1 sur la forme n2. n1 et n2 sont des nombres représentant des formes. n1 et n2 sont compris entre 0 et 15.

Exemple :

?COPIEFORME 0 1

Le papillon de la forme 0 est copié sur la forme 1.

COS n opération

(COSINUS). Rend le cosinus de n. n étant un angle exprimé en degrés. Voir aussi SIN.

Exemple :

```
?EC COS 45
0.707106782
?EC COS 90
0
```

Les autres fonctions trigonométriques peuvent être obtenues à partir de SIN et COS. La procédure suivante permet de définir la contangente d'un angle

```
?POUR COTANGENTE : ANGLE
>RENDS QUOT COS :ANGLE SIN : ANGLE
>FIN
```

CT commande

La commande CT (CacheTortue) cache la tortue, c'est-à-dire la rend invisible. Son état ainsi que son crayon n'en sont pas modifiés. L'exécution des procédures graphiques est alors plus rapide.

Exemple :

```
?AVANCE 50
?CT
?AVANCE 50
```

CTX	opération

(Couleur TeXte). Rend un nombre représentant le code de la couleur dutexte. Les codes de couleur pour le texte sur un écran couleur sont les suivants :

CODE	COULEUR
0	NOIR
1	ROUGE
2	VERT
3	JAUNE
4	BLEU
5	MAGENTA
6	CYAN
7	BLANC

CURS — opération

Rend une liste de 2 nombres, qui sont les coordonnées du curseur sur l'écran. Ces coordonnées sont comprises entre 0 et 24 pour les lignes et entre 0 et 39 pour les colonnes. Voir FCURS.

Exemple :

La procédure suivante permet d'obtenir une tabulation.

```
?POUR TAB
>TAPE CAR 32
>SI  (DER CURS -8 ) < 0 [ TAB]
>FIN


POUR TABNOTES
>TAPE "NOM TAB  EC "NOTE EC [ ]
>TAPE "MARTIN TAB EC 18
>TAPE "HERCULE  TAB EC 16
>TAPE "ALFRED  TAB EC 14.5
>FIN

?TABNOTES
NOM                                NOTE

MARTIN                             18
HERCULE                            16
ALFRED                             14.5
```

DEGELE commande

Redonne le mouvement aux tortues immobilisées par GELE.

DEMANDE tortue(s) liste   commande

Constitue provisoirement tortues en tortue(s) actives en leur appliquant les instructions contenues dans liste

Exemple :

```
? AV 20
? PARLE [ 0 1 ]
? CHACUN  [ REPETE 3 [ AV 30 TD 120 ] ]
? DEMANDE [ 2 3 ] [ REPETE 3 [ AV 50 TD 120 ] ]
```

.DEPOSE n1 n2 données — commande

Cette primitive LOGO permet d'écrire des octets (données) dans les différents segments mémoire (n1) à une adresse précise (n2).

n1 correspond au segment mémoire . Voir .BCHARGE
n2 correspond à l'adresse dans le segment

Exemple :

? .DEPOSE 0 12000 67

Ceci aura pour effet d'écrire dans le segment mémoire en RAM VDP à l'adresse 12000 (décimal) un octet dont la valeur est 67 .

DER obj — opération

Rend le DERnier élément de obj. Si obj est une liste ce sera le dernier élément de la liste (c'est à dire un mot ou une liste) et si obj et un mot, ce sera le dernier caractère du mot.

Exemple :

```
?ECRIS DER [ RAN TAN PLAN ]
PLAN
?ECRIS DER "RAN.TAN.PLAN
N
```

La procédure suivante permet d'écrire un mot ou une liste à l'envers :

```
?POUR ENVERS :OBJET
>SI VIDE? :OBJET [ STOP ]
>TAPE DER :OBJET
>ENVERS SD :OBJET
>FIN

?ENVERS "ENROUGE
EGUORNE
```

DETRUIS nom commande

Detruit le fichier nom sur le peripherique en cours (Cette commande n'exixte pas pour la cassette). Il ne faut pas utiliser la commande directe du systeme d'exploitation Exelmemoire ou disquette car elles risquent de detruire LOGO.

Exemple :

?DETRUIS "FORMES
Detruira le fichier FORMES qui se trouve dans le lecteur actif.

DIFF n1 n2 opération
n1-n2

(Différence). Rend le résultat de la soustraction de n2 à n1 .
Attention!
Le symbole des nombres négatifs. Il précède immédiatement un nombre (-5)
Le symbole de la soustraction: dans sa forme infixée il figure entre les deux données. (5 3)

Le moins unaire, suivi d'une donnée qui peut être un nombre positif ou négatif (si :X est 5, -:X rend -5)

Exemple:

?EC DIFF 85.56 19.713
65.967

?DONNE "N 12
?EC 30 - :N
12
?EC 30 - 8
22
EC 30 -8
30
QUE FAIRE DE -8

DIV n1 n2 opération
n1/n2

(DIVision) . Rend le résultat de la division de n1 par n2 (n2 ne peut pas être zéro)

Exemple:

?EC DIV 30 8
3.75
EC 30/8
3.75

DONNE nom obj commande

Donne le nom nom à obj qui peut être n'importe quel objet Logo, mot ou liste. Le statut de la donnée ainsi créée est celui d'une donnée globale sauf si l'affectation a lieu à l'intérieur d'une procédure faisant appel dans son titre à ce nom.

Exemple :

```
?DONNE "ANIMAL "MAMMIFERE
?EC :ANIMAL
MAMMIFERE

?DONNE :ANIMAL [BALEINE]
?ECRIS :MAMMIFERE
BALEINE

?ECRIS CHOSE :ANIMAL
BALEINE
?ECRIS COMPTE PREM CHOSE :ANIMAL
7
```

Car BALEINE a 7 éléments.

```
?DONNE "COTECARRE 80
?EC :COTECARRE
80
```

ECH opération

Rend une liste de deux nombres représentant l'échelle de l'écran graphique pour la tortue active. Voir FECH.

Exemple:

```
?VE
?EC ECH
100 100
```

ECRIS ,ECobj **commande**
(ECRIS,EC obj1 obj2....)

Affiche ses données et passe à la ligne. Les données sont affichées sur l'écran par défaut. Les listes sont affichées sans leurs crochets. Si ECRIS a plus d'une donnée, mettre le tout entre parenthèses. Voir aussi TAPE et MONTRE.

Exemple:

```
?ECRIS "BONJOUR
BONJOUR

?EC [BONJOUR A TOUS]
BONJOUR A TOUS

?EC [ ]

?EC "ALLO EC [QUI EST LA?]
ALLO
QUI EST LA?

?(EC "ALLO [QUI EST LA?])
ALLO QUI EST LA?
```

ED <u>proc</u> commande

Cette primitive permet d'entrer dans le mode éditeur. ED peut être suivi ou non d'un nom de procédure.

Exemple:

ED "CARRE

L'écran graphique rouge disparaît, il est remplacé par un écran texte de couleur cyan. En haut de cet écran le mot éditeur apparaît en cyan sur fond noir. Voir aussi l'éditeur de texte.

EDCAR n commande

Permet d'éditer le caractère n, n étant le code ASCII du caractère désiré (n doit être compris entre 32 et 127). Les touches permettant de se "déplacer" dans l'éditeur sont:
-Flèche haut
-Flèche bas
-Flèche droite
-Flèche gauche
-Barre d'espace pour remplir ou vider la case où se trouve le curseur.
Les touches <CTL> et <Q> permettent de valider le caractère.
Les touches <CTL> et <C> permettent de ne pas enregistrer une modification.

EDFORME n   commande

Fait entrer dans l'éditeur de formes et affiche la forme n, n étant un nombre allant de 0 à 15, représentant la forme à éditer. Les touches permettant de se déplacer dans l'éditeur sont:

-Flèche haut
-Flèche bas
-Flèche droite
-Flèche gauche
-Barre d'espace pour remplir ou vider la case où se trouve le curseur.
Les touches <CTL> et <Q> permettent de valider la forme.
Les touches <CTL> et <C> permettent de ne pas enregistrer une modification.

Exemple:

?EDFORME 0

Vous devez obtenir à l'écran la grille de 16 cases sur 16 cases représentant un papillon.

EGAL? obj1 obj2 opération
obj1 = obj2

Rend VRAI si obj1 et obj2 sont des nombres égaux des listes identiques ou des mots identiques, sinon rend FAUX. Dans sa forme infixée, obj1= obj2 les données doivent être écrites des deux cotés du signe égal (=).

Exemple :

```
?ECRIS EGAL? "A PREM PREM  [ ARBRE FLEUR ]
VRAI
?ECRIS EGAL? [BONJOUR ]  [BON [JOUR] ]
FAUX
?ECRIS EGAL?  "1E2   100
VRAI
```

La procédure suivante permet de savoir si deux objets Logo ont le même nombre d'éléments :

```
?POUR EGALITE :A  :B
>SI EGAL? COMPTE :A COMPTE  :B  [ECRIS [ MEME  NOMBRE
        D'ELEMENTS] ] [ECRIS [PAS LE MEME   NOMBRE D'ELEMENTS] ]
>FIN

?EGALITE "FRATERNITE  "LIBERTE
PAS LE MEME NOMBRE D'ELEMENTS
?EGALITE "AB   [A B]
MEME NOMBRE D'ELEMENTS
```

ENTREE n — commande

Cette commande LOGO permet de fixer le périphérique d'entrée. Ces périphériques d'entrées sont:
-Le clavier
-Ligne paralelle
-Ligne série
-L'éditeur

n doit être spécifié après la commande entrée. Les périphériques associés à n sont les suivants:

Code	périphérique
0	nul
1	clavier
2	ligne paralelle
3	ligne série
4	éditeur

EF nom(s) commande

Efface de l'espace de travail la (ou les) procédure(s) nom(s)

Exemple :

?EF "ELIZA
efface la procédure ELIZA.

?EF [MAISON CARRE TRIANGLE]
efface les procédures MAISON, CARRE, et TRIANGLE.

EFN nom commande

(EFface nom) . Efface de l'espace de travail le nom spécifié après la commande EFN .

Exemple :

?DONNE "ANNICK [ANTIBES]

?EFN "ANNICK

Ceci aura pour effet d'effacer de l'espace de travail le nom ANNICK .

EFNS commande

(EFface NomS) . Efface de l'espace de travail tous les noms qui ont été définis.

Exemple:

```
?DONNE "PATRICE [VALBONNE]
?DONNE "ALAIN [MANDELIEU]
?EFNS
```

EFNS aura pour effet d'effacer de l'espace de travail PATRICE et ALAIN.

EFPS commande

(EFface ProcédureS) . EFPS effacera de l'espace de travail toutes les procédures auparavant définies. EFPS n'effacera pas les noms. Comparez EFPS avec .EFTOUT.

Exemple:

```
?DOONE "PATRICE [JEAN]
?POUR TRUC
>REPETE 6 [ AV 20 TD 60 ]
>FIN
```

EFPS effacera la procédure TRUC mais n'effacera pas le nom PATRICE.

.EFTOUT	commande

(EFface TOUT) Précédée d'un point, cette commande efface de l'espace de travail tout ce qui a été créé.

ENTIER n — opération

Rend la partie entière de n en tronquant sa partie décimale. Voir aussi ARRONDIS.

Exemple:

```
?EC ENTIER 35.742
35
?EC ENTIER -1.5
-1
?EC ENTIER .5
0
```

La procédure suivante rend VRAI si la donnée est un nombre entier, FAUX sinon.

```
?POUR ENTIER? :N
>SI NON NOMBRE? :N [ RENDS [MAIS CE N'EST PAS UN NOMBRE ]]
>RENDS :N = ENTIER :N
>FIN

?EC ENTIER? 43
VRAI
?EC ENTIER 43/5
FAUX
?EC ENTIER? "UN
MAIS CE N'EST PAS UN NOMBRE
```

ESTAMPE commande

Imprime la copie de la forme de la tortue active sur l'écran. La couleur de l'estampe est celle du crayon et non celle de la tortue.

Exemple :

```
?CT
? REPETE 4 [ ESTAMPE AV 50 TD 90 ]
```

ET pred1 pred2 opération
(ET pred1 pred2 pred3)
(ET pred)

Rend VRAI si toutes ses données sont vraies simultanément, sinon rend FAUX. pred1, pred2.... est le résultat d'une opération qui rend le mot VRAI ou le mot FAUX. Si ET a plus ou moins que deux données, il faudra mettre les différents prédicats entre parenthèses.

Exemple :

?EC ET "VRAI "FAUX
FAUX
?EC ET EGAL? 5 (8-3) EGAL? (10+5) (7+8)
VRAI
?EC ET 5 6
ET N'AIME PAS 6 COMME DONNEE

.EXAMINE n1 n2 opération

Cette primitive LOGO permet d'examiner le contenu d'un octet. Les paramètres à transmettre sont n1 et n2.

n1 correspond au segment mémoire (Voir .BCHARGE et .BSAUVE)

n2 correspond à l'adresse située dans le segment mémoire.

Exemple :

?EC .EXAMINE 0 12000

LOGO retournera la valeur de l'octet situé dans la ram VDP à l'adresse 12000.

EXECUTE liste commande ou opération

Exécute la liste d'instructions comme si elle avait été tapée au clavier. Si liste est une opération, EXECUTE rend alors le résultat de liste.

Exemple :

```
POUR CALCULATEUR
>EC EXECUTE LL
>EC [ ]
>CALCULATEUR
>FIN

?CALCULATEUR
2 + 7
9
100 * 20
2000
```

HASARD n — opération

Rend un nombre entier positif, choisi au hasard entre 0 et (n-1). n doit être un nombre positif, s'il n'est pas entier, il sera arrondi à l'entier le plus proche.

Exemple:

HASARD 3 rend un nombre qui peut être 0, 1 ou 2

La procédure suivante rend un nombre au hasard pris dans un intervale spécifié.

```
?POUR PRENDS :LISTE
>DONNE "N1 PREM :LISTE
>DONNE "N2 DER :LISTE
>RENDS SOMME :N1 HASARD (1+:N2 - :N1)
>FIN
?EC PRENDS [-5 5]
2
```

LOGO	commande

Arrête complètement l'éxécution en cours et retourne au niveau supérieur. LOGO s'utilise à l'intérieur d'une procédure et a le même effet que les touches <CTL C>. Comparez avec STOP et RENDS.

Exemple:

```
?POUR DECOMPTER :N
>EC :N
>SI EGAL? :N  0 [ EC [FEU] [LOGO]
>DECOMPTER :N-1
>FIN


?POUR DEPART :N
>DECOMPTER :N
>EC [ C'EST PARTI]
>FIN

?DEPART 3
3
2
1
0
FEU
ARRET DANS DECOMPTER
```

Essayez les mêmes procédures en utilisant STOP à la place de LOGO.

FAUX	valeur

FAUX est une valeur rendue par LOGO. FAUX peut indiquer qu'une condition n'est pas remplie.

Exemple:

```
?EC MEMBRE? [ A] [ A B C]
FAUX
```

FCAP n commande

(Fixe CAP).Fixe la direction et le sens du déplacement de la tortue en degrés. Comme sur une boussole, le Nord (valeur 0) se trouve en haut de l'écran. Voir aussi CAP.

Exemple :

```
?VE
?FCAP 90
?AV 60
?VE
?FCAP 45
?AV 60
```

FCC n commande

(Fixe Couleur Crayon) Fixe la couleur du crayon de la tortue active à la couleur de code n. Voir CC pour les codes des couleurs.

Exemple:

?VE
?FCC 6
? EC CC
6

FCFG n commande

(Fixe Couleur Fond Graphique) Fixe la couleur du fond graphique à la couleur de code n. Voir aussi CF et CC pour les codes de couleur.

Exemple:

?FCFG 2
?EC CF
2

FCFORME n commande

(Fixe Couleur FORME) . Fixe la couleur de la tortue active à n. n étant un nombre compris entre 0 et 7 .

Exemple :

? FCFORME 2
? EC CFORME
? 2

FCT n commande

(Fixe Couleur Texte). Fixe la couleur des prochains caractères sur l'écran à n. Voir aussi CTX pour les codes des couleurs.

FCURS liste	commande

(Fixe CURSeur). Fixe le curseur à la position spécifiée par liste. liste comporte 2 nombres, le PREMier indique la ligne et le deuxième la colonne.
Les lignes sont numerotées de 0 à 24 et les colonnes de 0 à 39. Pourtant vous ne pouvez pas fixer le curseur à la dernière colonne car celle-ci est reservée à la flèche indiquant que la ligne Logo continue.

écran

0	39
24	

Exemple :

?FCURS [12 20] positionne le curseur au milieu de l'écran .
La procédure suivante permet de déplacer le curseur d'un certain nombre de lignes et de colonnes.

```
?POUR DEPLACEC :X :Y
>FCURS LISTE (:X + PREM CURS) (:Y + DER CURS)
>FIN
```

```
?ECRIS "0 DEPLACE 3  6 ECRIS "0
```

FECH liste commande

(Fixe ECHelle). liste comporte deux nombres compris entre 0 et 200 qui fixent respectivement les echelles horizontale et verticale de la tortue . Au démarrage, l'échelle est de [100 100].
FECH [50 100] a pour effet de diviser par deux les déplacements habituels de la tortue sur l'axe horizontal. Voir aussi ECH.

Exemple:

?FECH [50 100]
? FCAP 90
? AV 100

FFORME <u>n</u> commande

(Fixe FORME). Fixe la forme de la tortue active à la forme <u>n</u>. Au départ le numéro de la forme est zéro (cette forme correspond au papillon) .

Exemple :

? FFORME 1
? EC FORME
1

FFORME liste commande

FFORME peut être utilisée avec une liste de 4 nombres compris entre 0 et 15.Cette commande fixera la séquence des formes prises par la tortue grâce à la commande FVMARCHE.

Exemple:

?FFORME [0 1 0 1]
? EC FORME
0

FIN commande spéciale

Indique la fin de définition d'une procédure. FIN doit apparaître seul sur la dernière ligne de définition. Dès qu'une procédure a été définie, apparaît le message :

VOUS VENEZ DE DEFINIR....

et on est à nouveau au niveau supérieur, le symbole d'invite (?) et le curseur sont là.

Exemple :

```
?POUR CARRE  :COTE
>REPETE 4 [AVANCE  :COTE  TD 90]
>FIN
VOUS VENEZ DE DEFINIR CARRE
?
```

FLECTEUR n commande

(Fixe Lecteur). Fixe le lecteur **n** qui peut être le lecteur de disquettes , le magnétophone à cassettes, l'Exelmémoire en tant qu'uinté active.

0	Magnétophone à cassettes
1	Exelmémoire
2	Unité de disquettes A
3	Unité de disquettes B

Exemple:

?FLECTEUR 3

Le lecteur B devient l'unité active. Toutes les commandes concernant les fichiers iront vers le lecteur B sans que vous ayez besoin de le préciser.

?RAMENE "FORMES

Ramenera en mémoire le fichier appelé FORMES de la disquette qui se trouve dans le lecteur B.

Attention :

Il est conseillé d'écrire l'extension nom de fichiers qui vous indiquera à quel type appartient votre fichier. LOGO ne rajoute pas cette extension de nom de fichiers si celle-ci n'est pas précisé.

Exemple:

?SAUVE "FORMES.LOG

FORME	opération

Rend un nombre représentant la forme de la tortue active. S'il y a plusieurs tortues actives en même temps si par exemple vous avez fait PARLE [0 1 2 3] se sera la forme de la première tortue de la liste, c'est-à-dire la tortue 0. Voir FFORME .

Exemple:

? EC FORME
0

FPOS liste commande

(Fixe POSition). Fixe la position de la tortue sur l'écran à liste. liste contient deux nombres, la coordonnée X et la coordonnée Y d'une position. Notez la différence avec les déplacements par AVANCE et RECULE qui tiennent compte de la position et du cap actuel de la tortue. Le cap n'est pas modifié par la commande FPOS. La commande FPOS est indépendante de l'echelle. Voir aussi POS.

Exemple:

```
?VE
?FPOS [60 80]
?FPOS [-20 -80]
```

.FSERIE n1 n2 commande

Cette primitive permet d'initialiser la transmission série. On ne pourra effectuer de reception série qu'avec une machine Version 2 ou par l'emploi de la carte modem associée à la primitive AVEC.

La valeur de n1 varie entre 1 et 5. Elle permet la sélection de la valeur d'initialisation que l'on veut modifier. Si n1 est égal à 0, l'initialisation du port série sera réalisée avec les valeurs en cours.

n1 = 1

Permet l'initialisation de la longueur des mots, du nombre de bits stop, du mode de communication, du traitement de la parité, du protocole de communication ainsi que du mode de transmission.

n1 = 2

Permet de mettre le port série en mode de réception, transmission , transmission et réception ou off. Il permet également de "reseter" le flag d'erreur et le port série.

n1 = 3

Permet d'initialiser le mode horloge du circuit série, les modes de communications multiprocesseurs, la gestion de l'interruption 4 du 7041, et une partie du réglage de la vitesse de transmission.

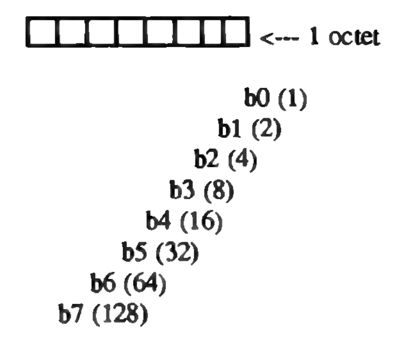
n1 = 4

Réglage de la vitesse de transmission

n1 = 5

Permet de positionner les niveaux des signaux DTR et DSR le l'interface série.

Le principe utilisé pour décrire la valeur à affecter à chaque octet est le suivant:



Comment positionner la valeur de n2?

n1 =1

b7	b6	b5 b4	b3 b2	b1	b0

b0:
- 0 Protocole MOTOROLA
- 1 Protocole INTEL

b1:
- 0 Comm isosynchrone
- 1 Comm asynchrone

b3 b2:
- 00 5 bits/mot
- 01 6 bits/mot
- 10 7 bits/mot
- 11 8 bits/mot

b5 b4:
- 00 Pas de traitement parité
- 01 parité impaire
- 11 parité paire

b6:
- 0 mode série
- 1 mode de communication

b7:
- 0 un bit stop
- 1 deux bits stop

b0. Protocole

Il existe deux protocoles possibles, l'intel et le motorola. Le positionnement à zéro de ce bit permet la sélection du mode MOTOROLA, la mise à 1 de ce bit sélectionne le mode INTEL. La communication multiprocesseurs diffère des autres modes de communication en utilisant les fonctions wake up et sleep. (Voir n=3) .

b1. Isosynchrone / Asynchrone

Ce bit détermine le mode de communication du port série. La mise à 1 de ce bit sélectionne le mode asynchrone.

b2.b3. Nombre de bits / mot

La longueur des caractères est programmable de 5 à 8 bits. Les caractères inférieurs à 8 bits sont complémentés par des zéros et cadrés à droite.

b4.b5. Contrôle de la parité

Si le bit 4 est à zéro, aucun contrôle ne sera effectué. Si le bit 4 est à 1, le contrôle de parité à réaliser se trouve décrit dans le bit 5. Ce dernier est à 1 pour une parité paire, à zéro pour une parité impaire.

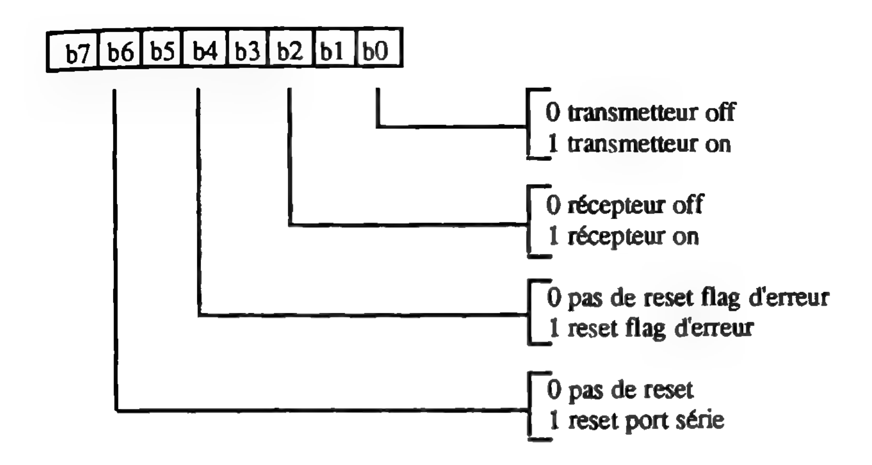
b6. Détermination mode série ou mode communication

Ce bit détermine si le port série opère en mode série ou dans un des deux modes de communication. La mise à zéro de ce bit sélectionne le mode série.

b7 Nombre de bits stop

Ce bit détermine le nombre de bits stop à émettre dans l'un des deux modes de communication. La mise à zéro de ce bit donne un bit stop, sa mise à 1 donne deux bits stop.

n1 = 2



b0. Transmetteur

La transmission des données ne peut s'effectuer que lorsque ce bit est à 1.

b2. Récepteur

L'interruption de réception n'est valide que si ce bit est à 1. Cependant, si ce bit est à zéro le registre à décalage continue d'assembler les caractères.

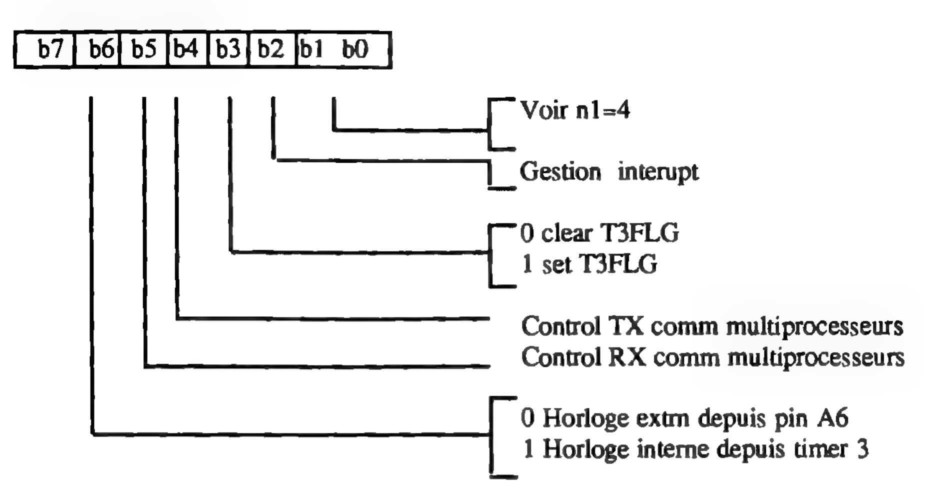
b4. Reset du flag d'erreur

La mise à 1 de ce bit provoque la réinitialisation des erreurs (à 0).

b6 Reset

La mise à 1 de ce bit provoque une condition de reset de l'UART.

n1=3



Seul b0 et b1 nous interesse . Voir n1 = 4

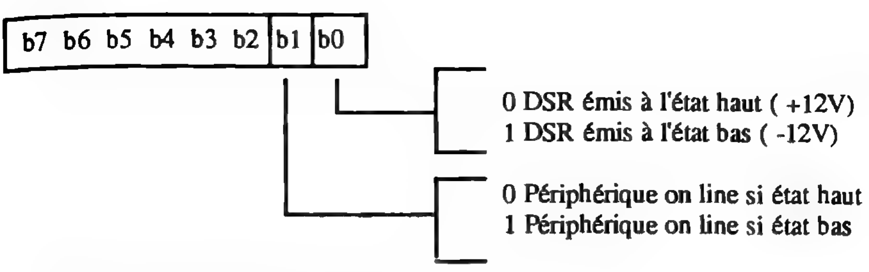
n1=4

b7	b6	b5	b4	b3	b2	b1	b0

La valeur de cet octet ainsi que la valeur indiquée dans b0 et b1 de l'octet précédant permettent de contrôler la vitesse de transmission.

b1 b0	octet n1=4	
01	174	110 bauds
00	255	150 bauds
00	127	300 bauds
00	63	600 bauds
00	31	1200 bauds
00	15	2400 bauds
00	7	4800 bauds
00	3	9600 bauds
00	1	19200 bauds
00	0	38400 bauds

n1 =5



A la mise sous tension, l'initialisation suivante est effectuée.

1200 bauds 7 bits / mot Parité paire 1 bit stop DSR, DTR état haut

Vous pouvez relire les valeurs d'initialisation par .SERIE

FVIT n commande

(Fixe VITesse). Fixe la vitesse de la tortue active à n, n étant un nombre allant de - 29999 à + 29999.

Exemple :

? LC
? FVIT 2
? VE
?FVIT 10

FVMARCHE n commande

Fixe la vitesse de succession des formes d'une tortue qui a reçu une liste de formes avec FFORME. FVMARCHE doit être utilisée avec FFORME et des primitives qui déplacent la tortue.

Exemple:

?FFORME [0 1 0 1]
?FVMARCHE 3
?FVIT 5

Attention: plusieurs formes ont été définies dans l'éditeur de formes grâce à la primitive EDFORME.

FX n commande

(Fixe X). Déplace la tortue horizontalement jusqu'au point n de la coordonnée X. L'origine est située au milieu de l'écran graphique. Voir aussi XCOR.

Exemple:

?VE
?FX 100
?FX 120

FY n — commande

(Fixe Y). Déplace la tortue verticalement jusqu'au point n de la coordonnée Y. La coordonnée X et le cap de la tortue n'en sont pas modifiés. Voir aussi YCOR .

Exemple:

```
?VE
?FY 100
```

GC commande

(GOMMECRAYON). Transforme le crayon de la tortue en gomme. Elle effacera toute trace rencontrée dans ses déplacements, par la couleur du fond graphique. Pour arrêter la gomme, il faut utiliser BC, LC .

Exemple:

```
?AV 150
?GC
?RE 50
```

La procédure CHRONO trace une "aiguille" qui tourne autours d'un point.

```
?POUR CHRONO
>CT LC
>REPETE 60 [ FPOS [ 0 0 ] BC AV 60 GC FPOS [ 0 0 ] AV 60 LC TD 6 ]
>FIN
```

GELE commande

Immobilise toutes les tortues actives. Elles reprendront leur mouvement par DEGELE.

IM <u>nom</u> commande

(IMprime) . Affiche la définition de <u>nom</u> .

Exemple:

```
?IM "CARRE
POUR CARRE
REPETE 4 [ AV 50 TD 90 ]
FIN

?DONNE "FORMES [ CARRE TRIANGLE]
?IM :FORMES
POUR CARRE
REPETE 4 [ AV 50 TD 90 ]
FIN

POUR TRIANGLE
REPETE 3 [ AV 50 TD 120 ]
FIN
```

IMNS commande

Affiche tous les noms de données existant dans l'espace de travail.

Exemple :

```
?IMNS
DONNE "COTE  100
DONNE "REPONSE "BALZAC
```

IMPS — commande

Affiche les définitions de toutes les procédures de l'espace de travail.

Exemple :

```
?IMPS
POUR CARRE
REPETE 4 [AV 50 TD 90]
FIN

POUR TRIANGLE
REPETE 3 [AV 50 TD 120]
FIN
```

IMTOUT commande

(IMprime TOUT) . Affiche tout ce qui est contenu dans l'espace de travail.

Exemple:

```
?IMTOUT
POUR CARRE :COTE
REPETE 4 [ AV :COTE TG 90]
FIN

POUR SPI :COTE :ANGLE
AV :COTE TG :ANGLE
SPI :COTE+8 :ANGLE
FIN

DONNE "COTE "100
```

IMTS commande

(IMprime TitreS) . Affiche les titres de toutes les procédures contenues dans l'espace de travail.

Exemple:

```
?IMTS
POUR CARRE :COTE
POUR SPI :COTE :ANGLE
```

ITEM n obj — opération

Rend le n-ième élément de obj. n doit être un entier positif, inférieur ou égal au nombre d'éléments de obj.

Exemple :

```
?ECRIS ITEM 2 [DO RE MI FA SOL]
RE
?ECRIS ITEM 3 "MUSIQUE
S
```

La procédure suivante permet d'insérer un mot d'une liste, à une place déterminée.

```
?POUR INSERE :MOT :N :LISTE
>SI :N =1 [RENDS MP  :MOT :LISTE]
>RENDS MP  PREM :LISTE INSERE :MOT :N  -  1  SP :LISTE
>FIN

?ECRIS INSERE  "MI  3  [DO RE  FA SOL ]
DO RE MI FA SOL
```

LC	commande

Lève le crayon de la tortue. Elle ne laissera plus la trace de son déplacement. Pour pouvoir tracer à nouveau, il faut utiliser BC.

Exemple:

?AVANCE 50
?LC
?AVANCE 50

LECTEUR opération

Rend le numéro du périphérique de stockage actif. Par défaut le lecteur est 0. Voir aussi FLECTEUR.

Exemple:

?EC LECTEUR
1
L'Exelmémoire est le périphérique courant.

LISTE obj1 obj2 opération

Rend la liste dont les éléments sont obj1, obj2 . Voir aussi PHRASE.

Exemple :

?MONTRE LISTE "AFRIQUE [EUROPE ASIE]
[AFRIQUE [EUROPE ASIE]]

?MONTRE LISTE [AU CLAIR] [DE LA LUNE]
[[AU CLAIR] [DE LA LUNE]]

LISTE? obj opération

Rend VRAI si obj est une liste, sinon rend FAUX.

Exemple :

?ECRIS LISTE? "TROIS
FAUX

?ECRIS LISTE? [TROIS]
VRAI

?ECRIS LISTE? SP [TROIS]
VRAI

La procédure suivante permet de savoir si une donnée est un mot ou une liste.

?POUR SAVOIR :OBJET
>SI LISTE? :OBJET [ECRIS [CECI EST UNE LISTE]] [ECRIS [CECI EST UN MOT]]
>FIN

?SAVOIR [AN AS TAZ]
CECI EST UNE LISTE

LL	opération

(LISLISTE) . Rend sous forme de liste la prochaine ligne lue à partir du clavier ou de l'entrée courante dès que vous appuyez sur la touche <ENTREE> . Ce que vous tapez est affiché à l'écran.

Exemple:

```
?MONTRE LL
BONJOUR LOGO
[BONJOUR LOGO]

?POUR DIALOGUE
>EC [ BONJOUR, COMMENT T'APPELLES TU? ]
>DONNE "REPONSE  LL
>EC PHRASE "BONJOUR :REPONSE
>FIN

?DIALOGUE
BONJOUR COMMENT  T'APPELLES TU?
HECTOR
BONJOUR HECTOR
```

LISCAR opération

Rend le prochain caractère lu à partir du clavier ou de l'entrée courante. Si aucun caractère n'est tapé, LISCAR attend jusqu'à ce que vous tapiez quelque chose. Notez que LISCAR n'affiche pas à l'écran ce que vous tapez au clavier.

Exemple :

La procédure suivante déplace la tortue en appuyant sur une touche.

A pour AVANCE 5, R pour RECULE 5, D pour TD 10.

```
?POUR CONDUIRE
>DONNE "CAR LISCAR
>SI :CAR = "A [AVANCE 5]
>SI :CAR = "R [RECULE 5]
>SI :CAR = "D [TD 10]
>CONDUIRE
>FIN
```

MANETTE n opération

Rend un entier compris entre 128 et 131, représentant la rotation du levier de la manette n. (n étant 0 ,1)

Exemple :

La procédure suivante permet de dessiner sur l'écran en utilisant les manettes:

```
?POUR DESSINER
>TD MANETTE 0
>AV MANETTE 1
>DESSINER
>FIN
```

MD obj liste opération

(Mets en dernier). Rend une nouvelle liste, formée de liste à laquelle on a mis obj en DERnier élément. Comparer avec MP.

Exemple :

?MONTRE MD [BLEU] [BLANC ROUGE]
[BLANC ROUGE [BLEU]]

?MONTRE MD [BLEU BLANC] [ROUGE]
[ROUGE [BLEU BLANC]]

ME n1 n2 commande

(Mixe Ecran). Redéfini l'écran graphique avec une partie pour le graphique et une partie pour le texte. Avec ME la page graphique débute à la ligne n1 et comporte n2 lignes.

Exemple:

?ME 4 10

ME 4 10 aura pour effet d'avoir un écran graphique commençant à la ligne 4 et s'étendant sur 10 lignes.

MEMBRE? <u>obj1</u> <u>liste</u> opération

Rend VRAI si <u>obj</u> (caractère, mot ou liste) est un élément de <u>liste</u>; sinon rend FAUX. Note : un caractère est l'élément d'un mot, un mot est l'élément d'une liste, et une liste l'élément d'une liste de listes.

Attention! le mot vide n'est pas un élément de la liste vide.

Exemple :

```
?ECRIS MEMBRE? [A B] [A B C]
FAUX

?ECRIS MEMBRE? [A B ] [ [A B ] C]
VRAI
```

La procédure suivante permet de reconnaître si une donnée est une voyelle :

```
POUR VOYELLE? :LETTRE
>RENDS MEMBRE? :LETTRE [A E I O U Y a e i o u y]
>FIN

?ECRIS VOYELLE? PREM SP "RAT
VRAI
```

MONTRE obj	commande

Affiche obj et passe à la ligne, en "montrant" sa nature (les listes apparaissent avec les crochets). Comparez avec TAPE et ECRIS.

Exemple :

```
?MONTRE "A
A
?MONTRE [UNE LISTE]
[UNE LISTE]
```

MOT mot1 mot2 opération

Rend le mot obtenu en soudant ses données.

Exemple :

```
?ECRIS MOT "PAR "TERRE
PARTERRE

?ECRIT MOT "PAR [TERRE]
MOT N'AIME PAS [TERRE] COMME DONNEE
```

La procédure suivante rend un mot avec le préfixe PARA.

```
POUR PREFIXER :MOT
>RENDS MOT "PARA :MOT
FIN

?ECRIS PREFIXER "PLUIE
PARAPLUIE
```

MOT? obj　　　　　　　　　　　　　opération

Rend VRAI si obj est un mot, sinon rend FAUX.

Exemple :

```
?ECRIS MOT? "RAT
VRAI

?ECRIS MOT? [RAT]
FAUX

?ECRIS MOT? SAUFPREMIER [VERT]
FAUX

?ECRIS MOT? PREMIER [VERT]
VRAI
```

La procédure suivante détermine si sa donnée est un mot ou une liste. Si la donnée est un mot, elle teste pour voir si c'est un nombre.

```
POUR DEVINE :DONNEE
>SI MOT? :DONNEE [SI NOMBRE? :DONNEE [ECRIS [C'EST UN NOMBRE] ]
  [ECRIS [C'EST UN MOT] ] ] [ECRIS [C'EST UNE LISTE] ]
>FIN

?DEVINE 8
C'EST UN NOMBRE

?DEVINE "HUIT
C'EST UN MOT

?DEVINE [HUIT]
C'EST UNE LISTE
```

MP obj liste opération

(Mets en Premier). Rend une nouvelle liste, formée de liste à laquelle on a mis obj en premier élément. Comparer avec MD.

Exemple :

```
?MONTRE MP  [BLEU] [BLANC ROUGE]
[ [BLEU] BLANC ROUGE]

?MONTRE MP [BLEU BLANC] [ROUGE]
[ [BLEU BLANC] ROUGE]
```

La procédure suivante permet d'avoir un dictionnaire de fleurs.

```
?POUR BOTANIQUE
>TAPE  [RAJOUTE UN NOM DE FLEUR A LA LISTE :]
>ECRIS :FLEURS
>DONNE "NOUVELLE LL
>DONNE "FLEURS MP  :NOUVELLE :FLEURS
>ECRIS [ VOICI TOUTES LES FLEURS QUE JE CONNAIS]
>ECRIS :FLEURS
>FIN


?DONNE "FLEURS [ROSE  MARGUERITE]
?BOTANIQUE
RAJOUTE UN NOM DE FLEUR A LA LISTE :
ROSE  MARGUERITE
THERESE
VOICI TOUTES LES FLEURS QUE JE CONNAIS
THERESE ROSE  MARGUERITE
```

MT commande

Montre la tortue sur l'écran. Voir aussi CT (cachetortue) , et VISIBLE?.

Exemple:

```
?AVANCE 50
?CT
?MT
```

NETTOIE commande

Efface l'écran graphique avec la couleur de fond graphique courante sans modifier ni la position ni l'orientation de la tortue. Comparer avec VE (VIDECRAN) .

Exemple:

```
?BC
?AV 60
?TD 75
?AV 60
?NETTOIE
```

NOM? mot opération

Rend VRAI si mot désigne un objet Logo, c'est à dire si :mot existe, sinon rend FAUX.

Exemple :

?EC NOM? "ELEVE
FAUX

?DONNE "ELEVE "SYLVAIN
?EC NOM? "ELEVE
VRAI

?EC :ELEVE
SYLVAIN

NOMBRE? obj opération

Rend VRAI si obj est un nombre, sinon rend FAUX.

Exemple :

```
?ECRIS NOMBRE? "TROIS
FAUX

?ECRIS NOMBRE? 3
VRAI

?ECRIS NOMBRE? [3]
FAUX
```

La procédure suivante rend VRAI si sa donnée est une liste de nombres :

```
POUR LISTENB? :LISTE
>SI :LISTE = [ ] [ RENDS "FAUX]
>SI (COMPTE :LISTE) = 1 [RENDS NOMBRE? PREM  :LISTE]
>RENDS ET NOMBRE? PREM :LISTE LISTENB? SP  :LISTE
>FIN

?ECRIS LISTENB? [5 5 5]
VRAI
```

NON pred — opération

Rend la valeur logique opposée à celle de pred. Si pred est vrai, rend FAUX et si pred est faux, rend VRAI.

Exemple :

```
?EC NON EGAL? (8+3) 20
VRAI
```

ORIGINE commande

Ramène la ou les tortues au centre de l'écran, cap à 0. Cette commande équivaut à FPOS [0 0] et FCAP 0. Même si le crayon de la tortue est baissé ORIGINE ne laissera pas la trace de son passage.

Exemple:

```
?AV 60
?TD 75
?AV 60
?ORIGINE
```

OU pred1 pred2 opération
(OU pred1 pred2 pred3...)
(OU pred1)

Rend FAUX si toutes ses données sont simultanément fausses; sinon rend VRAI. Si OU a plus ou moins que deux données, inclure le tout entre parenthèses.

Exemple :

```
?EC OU EGAL? 1 2 EGAL? "A PREM "ATTENTION
VRAI
```

PARLE <u>tortue(s)</u> commande

Fait de <u>tortue(s)</u> la (les) tortue(s) active(s) et les prochaines instructions données leur seront adressées.

Exemple :

?PARLE 1
?EC QUI
1
? PARLE [0 1 2 3]
? EC QUI
0 1 2 3
Quatre tortues sont activées. Voir DEMANDE et CHACUN .

? REPETE 6 [AV 30 TD 60]

Seule la tortue 0 est concernée par cette séquence. Si vous désirez que les quatres tortues effectuent cette séquence il faut utiliser CHACUN.

PHOTO commande

Remplace la forme de la tortue active, par les points (n'ayant pas la couleur du fond) se trouvant sous la tortue. Comparer avec ESTAMPE.

Exemple:

```
?VE
?REPETE 4 [ AV 5 TD 9O ]
?PHOTO
?LC
? AV 45
```

Le papillon de la forme 0 a été remplacé par un carré.

PHRASE obj1 obj2
PH obj1 obj2 — opération

Rend une liste construite à partir de ses données. Voir aussi LISTE .

Exemple :

La procédure suivante initie un dialogue avec l'ordinateur :

```
POUR DIALOGUER
>ECRIS [BONJOUR COMMENT VOUS APPELEZ  VOUS?]
>DONNE "REPONSE LL
>ECRIS PHRASE "BONJOUR PREM :REPONSE
>FIN

?DIALOGUER
BONJOUR COMMENT VOUS APPELEZ VOUS?
HIPPOLYTE MONPOUX
BONJOUR HIPPOLYTE
```

PLG? n1 n2 opération
n1>n2

(PLus Grand?). Rend VRAI si n1 est un nombre supérieur à n2. Sinon rend FAUX. L'opération infixée, >, s'écrit entre les deux nombres.

Exemple :

```
?EC PLG? 2 5
FAUX

?EC -8 > -10
VRAI
```

La procédure suivante rend VRAI si la deuxième donnée est supérieure aux deux autres.

```
?POUR ENDEHORS :A :B :C
>RENDS ET :B>:A :B> :C
>FIN

?EC ENDEHORS 3 5 4
VRAI
```

PLP? n1 n2 opération
n1>n2

(PLus Petit?) . Rend VRAI si n1 est un nombre inférieur à n2 . Sinon rend FAUX . L'opération infixée, >, s'écrit entre les deux nombres.

Exemple:

```
?EC PLP? 2 5
VRAI
```

```
?EC -8 > -10
FAUX
```

La procédure suivante rend VRAI si la deuxième donnée est un nombre situé entre les deux autres.

```
?POUR ENTRE :A :B :C
>RENDS ET :A < :B  :B< :A
>FIN
```

```
?EC ENTRE 7 8 9
VRAI
```

POINT liste commande

Affiche un point à la position spécifiée par liste, sans faire bouger la tortue. liste comporte deux nombres représentant la coordonnée horizontale et la coordonnée verticale du point. La couleur du point est celle du crayon de la tortue.

Exemple :

?POINT [20 80]

POS opération

(POSition). Rend la position actuelle de la tortue sous forme de liste de deux nombres : la coordonnée horizontale et la coordonnée verticale.

Exemple :

?VE
?AV 100
?EC POS
0 100

POUR nom commande
POUR nom donnée1 donnée2

Débute la définition d'une procédure appelée nom avec, s'il y a lieu, des données.

Chaque nom de donnée doit être précédé de deux points (:).

Quand POUR est utilisé en mode direct, le symbole d'invite (?) change à la prochaine ligne en (>) pour vous rappeler que vous êtes en mode de définition de procédures : les instructions tapées ne sont pas executées mais mémorisées.

FIN termine la définition et permet de retrouver le mode direct.

Attention! nom ne doit pas être un nombre, le nom d'une primitive ou le nom d'une procédure déjà définie.

Exemple :

```
?POUR CARRE :COTE
>REPETE 4 [AV :COTE TD 90]
>FIN
VOUS VENEZ DE DEFINIR CARRE
?
```

PREM obj opération

Rend le premier élément de obj. Si obj est une liste ce sera le PREM élément de la liste (c'est-à-dire un mot ou une liste). Si obj est un mot, ce sera le PREM caractère du mot. PREM n'accepte ni le mot vide, ni la liste vide comme donnée.

Exemple :

```
?MONTRE PREM [RAN TAN  PLAN ]
RAN

?MONTRE PREM .35
0
```

La procédure suivante permet de calculer la position d'une lettre dans un mot pour sa première occurence :

```
?POUR OCC :LETTRE :MOT
>SI VIDE? :MOT [RENDS PHRASE :LETTRE [N'EXISTE PAS DANS CE MOT] ]
>SI EGAL? PREM :MOT :LETTRE [RENDS 1] [RENDS SOMME 1 OCC :LETTRE SP :MOT
>FIN


?ECRIS OCC "E  "RHINOCEROS
7
```

PRIM commande

Affiche les primitives d'Exelogo sur la sortie courante fixée par la commande SORTIE. La sortie courante prise par défaut est l'écran.

PRIM? mot opération

Rend VRAI si mot est le nom d'une primitive du langage Logo, sinon rend FAUX.

Exemple :

?ECRIS PRIM? "AVANCE
VRAI

?ECRIS PRIM? "CARRE
FAUX

PROC? mot opération

Rend VRAI si mot est le nom d'une procédure définie; sinon rend FAUX.

Exemple:

```
?EC PROC? "CARRE
VRAI
```

PROD n1 n2 opération
n1 * n2

(PRODuit). Rend le produit de ses données.

Exemple:

```
?EC PROD 5 7
35
?EC 5*9
45
?EC PROD 5 SOMME 7 3
50
```

La procédure suivante rend le carré d'un nombre.

```
?POUR CARRE :X
>RENDS PROD :X :X
>FIN

?EC CARRE 5
25
```

QUI   opération

Rend le code de la ou des tortues actives à un moment donné.

Exemple :

```
? PARLE [ 0 1 2 3 ]
? EC QUI
0 1 2 3
```

QUOT n1 n2 opération

(QUOTient). Rend le quotient entier de la division de n1 par n2 . (n2 ne peut être égal à zéro)

Exemple:

?EC QUOT 17 13
1

RAMENE nomfichier commande

Ramène en mémoire centrale le contenu du fichier nomfichier Voir aussi SAUVE.

Exemple :

?RAMENE "FORMES.LOG
Ramène en mémoire le fichier FORMES.LOG

RAMENECAR nomfichier commande

Cette primitive permet de charger en mémoire le générateur de caractères préalablement sauvergardé par SAUVECAR.

Exemple:

?RAMENECAR "GRECQUE.CAR

RAMENEFORME nomfichier commande

Cette primitive permet de charger en mémoire les formes sauvegardées à l'aide de SAUVEFORME .

Exemple:

?RAMENEFORME "GEO.LOG

RAMENEIMAGE nomfichier commande

Permet de charger une image à l'écran sauvée par SAUVEIMAGE.

RC n opération

(Racine Carrée) . Rend la racine carrée de n .

Exemple:

?EC RC 16
4

?EC RC 753
27.44084547

RE n commande
RECULE n

Fait reculer la tortue dans la direction en cours de n pas, en tenant compte de l'echelle actuelle. Voir FECH .

Exemple :

?VE
?RE 200

RECYCLE commande

Effectue un "nettoyage" pour libérer autant de place mémoire que possible.

Ce nettoyage se fait automatiquement quand nécessaire, mais prend au moins une seconde.

Il est utile de faire RECYCLE avant d'exécuter un long programme car ceci permet d'avoir un temps régulier d'exécution. Ainsi ous évitez que le nettoyage se fasse pendant l'exécution du programme.

REMPLIS commande

Remplace la couleur sur laquelle se trouve le crayon par la couleur du crayon, pour une surface limitée par un tracé de couleur différente .

Exemple:

?REPETE 36 [AV 10 TD 10]
?LC
?FX 20
?FCC 3
?REMPLIS

RENDS obj commande

Termine l'exécution de la procédure et fait de obj ce qui est rendu par cette procédure. obj sera alors la donnée de la procédure appelante. RENDS s'utilise à l'intérieur d'une procédure. Comparer avec STOP et LOGO.

Exemple :

ABS rend la valeur absolue d'un nombre.

```
POUR ABS :N
>SI :N < 0 [RENDS -:N ] [RENDS :N ]
>FIN

?EC ABS 3
3
?EC ABS -3
3
```

REPETE n liste commande

Exécute la liste d'instructions n fois. Si n est négatif il y a erreur; si n est décimal il est arrondi.
Attention! Une liste d'instructions, tout comme une ligne Logo doit être une suite de commandes.

Exemple :

?REPETE 360 [AV 1 TD 1]
Trace un cercle.

?REPETE 3 [AV 100 TG 120]
Trace un triangle équilatéral de 100 pas de côté.

.... Dans une classe moderne!

```
POUR PUNITION
>REPETE 10 [ECRIS [JE NE PARLERAI PLUS AVEC MON VOISIN] ]
>FIN
```

SAUVE nomfichier commande

Sauve tout le contenu de l'espace de travail dans le fichier nomfichier dans le périphérique de stockage courant. Les noms de fichiers de plus de 13 caractères sont tronqués pour le périphérique cassette.

SAUVECAR nomfichier commande

Cette primitive permet de sauvegarder les caractères (générateur de caractères) que vous avez défini à l'aide de EDCAR. Voir EDCAR.

Exemple:

?SAUVECAR "GRECQUE.CAR

*Notez que nous avons donné l'extension .CAR pour se rappeler que nous venons de sauver un générateur de caractères.

SAUVED nomfichier commande

Sauve le contenu de l'éditeur, quel qu'il soit dans le fichier nomfichier .

SAUVEFORME nomfichier commande

Cette primitive permet de sauver les formes que vous aurez définies par FFORME.

Exemple:

?SAUVEFORME "GEO.LOG

SAUVEIMAGE <u>nomfichier</u> commande

Cette primitive permet de sauvegarder une image réalisée à partir de LOGO. Attention seule l'image sera sauvegardée .

Exemple :

Vous avez défini une procédure qui trace des carrés sur l'écran.

```
?POUR CARRE
>REPETE 4 [ AV 50 TD 90 ]
>FIN
```

```
? CARRE
```

Exécution de la procédure

```
?SAUVEIMAGE "CARRE.DES
```

Ceci aura pour effet de sauvegarder l'image qui se trouve sur votre écran . La procédure CARRE n'est pas sauvegardée. Nous avons pris soin de spécifier une extension de nom de fichier (.DES) . Cette extension .DES est très utile si vous désirez traiter des images réalisées avec LOGO par le logiciel EXELPAINT.

SD obj opération

Rend obj sans son DERnier élément. Si obj est une liste ce sera la liste sans son DERnier élément (mot ou liste) et si obj est un mot ce sera le mot sans son DERnier caractère. SD n'accepte ni le mot vide ni la liste vide comme donnée.

Exemple :

```
?MONTRE SD  [RAN TAN PLAN ]
[RAN  TAN]

?MONTRE SD  "RAN.TAN.PLAN
RAN.TAN.PLA

?MONTRE SD   .123
.12
```

Pour trouver la racine d'un verbe régulier donné à l'infinitif.

```
?POUR RACINE :VERBE
>RENDS SD SD :VERBE
>FIN

?ECRIS RACINE "PARLER
PARL
```

.SERIE	opération

Cette opération permet de relire les valeurs d'initialisation fixées par la commande .FSERIE (Voir aussi FSERIE)

SI préd liste1 commande ou opération
SI préd liste1 liste2

Si préd est VRAI, la liste d'instructions liste1 est exécutée. Si préd est FAUX la liste d'instructions liste2 (s'il y a lieu) est exécutée. Si liste2 n'existe pas, la procédure continue avec la prochaine ligne.

Attention! SI avec toutes ses données doit être écrit sur une même ligne Logo. Il est conseillé d'écrire des paranthèses autour de préd.

Exemple :

```
POUR DECIDE
>SI  HASARD 2  = 0 [RENDS "BONJOUR]
>RENDS [AU REVOIR]
>FIN
```

La même procédure peut être écrite de la façon suivante :

```
POUR DECIDE
>SI (HASARD 2)  = 0 [RENDS "BONJOUR] [RENDS [AU REVOIR] ]
>FIN
```

Ou encore de la façon suivante :

```
POUR DECIDE
>RENDS SI (HASARD 2)  = 0 ["BONJOUR] [ [AU REVOIR] ]
>FIN
```

Notez le double crochet de AU REVOIR le premier indique la liste donnée de SI et le deuxième la liste-objet Logo.

SIN n opération

(SINus). Rend le sinus de n. n étant un angle exprimé en degrés. Voir aussi COS.

Exemple:

?EC SIN 45
0.707106812

SOMME n1 n2 opération
n1 + n2

Rend la somme de ses données.

Exemple:

?EC SOMME 3 90
93

?EC 7 + -5
2

SORTIE n commande

Cette commande LOGO permet de fixer le périphérique de sortie . Voir aussi ENTREE.
Les périphériques associés à n sont les suivants.

Code	périphérique
0	nul
1	moniteur/écran
2	ligne paralelle
3	ligne série
4	éditeur

SP obj — opération

Rend obj sans son PREMier élément. Si obj est une liste ce sera la liste sans son PREMier élément (mot ou liste) et si obj est un mot, ce sera le mot sans son PREMier caractère. SP n'accepte ni le mot vide ni la liste vide .

Exemple :

```
?MONTRE SP [ RAN TAN PLAN ]
[ TAN PLAN ]

?MONTRE SP "RAN.TAN.PLAN
AN.TAN.PLAN

?MONTRE SP .123
.123
```

La procédure suivante permet d'enlever un mot d'une liste.

```
?POUR ENLEVE :MOT :LISTE
>SI VIDE? :LISTE [RENDS [ ]]
>SI :MOT = PREM :LISTE [RENDS SP :LISTE]
>RENDS PHRASE PREM :LISTE ENLEVE :MOT SP :LISTE
>FIN

?ECRIS ENLEVE "TA [TA RA MA]
RA MA
```

STOP commande

Arrête la procédure en cours et rend le contrôle à la procédure appelante ou au niveau supérieur. STOP n'a de sens qu'à l'intérieur d'une procédure.

Une procédure qui contient STOP est une procédure de type commande. Comparez av RENDS et LOGO.

Exemple :

```
POUR DECOMPTER :N
>EC :N
>SI :N = 0 [EC [FEU! ] EC CAR 7 STOP]
>DECOMPTER :N - 1
>FIN


?DECOMPTER 3
3
2
1
0
FEU!
```

SURCOULEUR opération

Rend le code de la couleur de ce qui se trouve sous le crayon de la tortue active, même si le crayon est levé.

Exemple:

```
? VE
? EC SURCOULEUR
1
```

.SYSTEME commande

Passe le contrôle de la machine au système d'exploitation du périphérique concerné (sauf la cassette). Cette commande permet de consulter le répertoire des fichiers présents. Il est tres dangereux de donner une commande "externe" au système d'exploitation de disquettes ou une commande au système de l'Exelmémoire. Cette commande aurait de grandes chances de détruire l'espace de travail de LOGO, et au retour (hypothètique) vous auriez perdu vos procedures, définitions, noms.... De même, il ne faut jamais éteindre la machine ou appuyer sur le bouton <RESET> quand vous êtes sous le système d'exploitation Exelmémoire, car vous perdriez au moins la moitié de l'espace disponible sur ce périphérique, que LOGO s'attribue temporairement.

Pour revenir depuis le système d'exploitation Exelmémoire vers LOGO, taper <ESC>

Pour revenir vers LOGO depuis le DOS, taper EXIT puis <ENTREE>

TAN n opération

(Tangente) . Rend la tangente de l'angle en degrés.

Exemple :

?EC TAN 45
1

TAPE obj — opération
(TAPE obj1 obj2 .....)

Affiche ses données mais sans passer à la ligne. Les liste sont affichées sans les crochets. Si TAPE a plus d'une donnée, inclure tout entre parenthèses. Comparez avec ECRIS et MONTRE.

Exemple:

```
?TAPE "A
A
?TAPE [A B C]
A B C ?(TAPE "A [A B C])
AA B C?


?POUR SIGLE :MOT
>SI VIDE?  :MOT [EC []  STOP]
>TAPE PREM :MOT TAPE [.]
>SIGLE SP :MOT
>FIN

?SIGLE  "CGCT
C.G.C.T.
```

TAPISSE commande

Remplit un région fermée de l'écran, (ou tout l'écran au cas où il n'y aurait pas de région fermée) avec des copies de la forme de la tortue active. SURCOULEUR doit être différent de la couleur du crayon et de la couleur de la forme de la tortue.

Exemple :

```
? VE
? FCC 3
? FCFORME 4
? REPETE 6 [ AV 30 TD 60 ]
? LC
? FCAP 90
? AV 20
? TAPISSE
```

TD n commande

(TOURNEDROITE). Fait pivoter la tortue sur sa droite de **n** degrès (dans le sens des aiguilles d'une montre). La commande TD n augmente la valeur du cap de n.

Exemple :

```
?VE
?TD  60
?AV 50
?EC CAP
60
```

TE commande

(Texte Ecran).Consacre tout l'écran au texte. Le graphique sera invisible jusqu'à ce qu'une procédure graphique soit exécutée.

TG n commande

(TOURNEGAUCHE). Fait pivoter la tortue sur sa gauche de n degrés (dans le sens inverse des aiguilles d'une montre). La commande TG n diminue la valeur du cap de n.

Exemple:

```
?VE
?TG 60
?AV 50
?EC CAP
300
```

TOUCHE? opération

Rend VRAI si au moins un caractère attend d'être lu depuis le clavier ou l'entrée courante, et n'a pas encore été lu par LISCAR ou LL sinon rend FAUX.

Exemple :

```
?POUR VIRAGE
>AV 2
>SI TOUCHE? [TOURNE LISCAR]
>VIRAGE
>FIN

?POUR TOURNE :DIR
>SI :DIR = "D [TD 10]
>SI :DIR = "G [TG 10]
>FIN
```

La tortue se déplace continuellement dans une direction jusqu'à ce que vous appuyez sur D ou G. En appuyant sur une seule touche, vous pouvez ainsi faire des dessins.

VE	commande

(VIDECRAN). Efface l'écran graphique et replace la tortue à l'origine. Comparer à NETTOIE et ORIGINE .

Exemple :

```
?AVANCE 60
?TD   75
?AVANCE 60
?VE
```

VIDE? obj — opération

Rend VRAI si obj est le mot vide ou la liste vide sinon rend FAUX.

Exemple :

```
?ECRIS VIDE? 0
FAUX

?ECRIS VIDE? SP 0
VRAI
```

La procédure suivante fait une correspondance élément par élément entre 2 listes.

```
POUR CORRESPONDRE :UNELISTE  :UNEAUTRE
>SI OU VIDE?  :UNEAUTRE VIDE? :UNELISTE [ECRIS [IL N'Y A  PLUS
  D'ELEMENT ] STOP]
>ECRIS PHRASE PREM :UNELISTE PREM  :UNEAUTRE
>CORRESPONDRE SP :UNELISTE SP :UNEAUTRE
>FIN

?CORRESPONDRE [1 2 3] [UN DEUX TROIS]
1 UN
2 DEUX
3 TROIS
IL N'Y A PLUS D'ELEMENTS
```

VISIBLE? opération

Retourne VRAI si la tortue est visible; sinon rend FAUX.

exemple :

```
?AVANCE 50
?CT
?EC VISIBLE?
FAUX
```

VIT opération

Rend un nombre représentant la vitesse actuelle de la tortue active.

Exemple :

?FVIT 2
?EC VIT
2

VMARCHE opération

Rend un nombre représentant la vitesse de succession des formes fixée dans FVMARCHE.

Exemple:

?EC VMARCHE
3

VRAI valeur

VRAI est une valeur rendue par LOGO. VRAI peut indiquer qu'une condition est remplie.

Exemple:

?EC NOMBRE? 45
VRAI

En effet 45 est un nombre.

?EC MEMBRE? [A] [[A] B C]
VRAI

VT	commande

Vide le texte de l'écran et place le curseur au début de la première ligne de l'écran texte.

XCOR opération

Rend la coordonnée X de la position actuelle de la tortue.

Exemple:

```
?VE
?EC XCOR
0

?TD 90
?AVANCE 50
?EC  XCOR
50
```

YCOR — opération

Rend la coordonnée Y de la position actuelle de la tortue.

Exemple :

```
?VE
?AV 70
?EC YCOR
70
```